The Interpretation of Tables in Texts

Matthew Francis Hurst

PhD.

The University of Edinburgh

2000

I declare that

1. I am the author of this thesis

2. The work reported in this thesis is entirely my own unless otherwise attributed.

Matthew Francis Hurst

Acknowledgements

I would like to thank the staff, my colleagues and friends at the University of Edinburgh who have helped in anyway with the research reported in this thesis. In particular I would like to thank the Language Technology Group of the Human Communication Research Centre for providing me with a wonderful environment for research. The LTG is an excellent group and I feel honoured to have been a member. Shona Douglas has been a great help in getting the research off the ground and in getting my initial funding, with the support of George Imlah of BICC.

I would also like to thank the members of my second research home in Japan: the Electrotechnical Laboratory. It was due to the kind invitation of Toshi Kato that I was able to visit Tsukuba and enjoy a very different and exciting place to carry out some of my research. I am indebted to all the researchers and staff there. The European Commission's Science and Technology Fellowship programme must also be thanked for funding the visit.

Thanks must also go to those who read several drafts of various parts of the thesis. My supervisors Ewan Klein and Marc Moens gave invaluable comments. Special thanks go also to Janet Hitzeman for not only reading several drafts but also carrying them to various far flung locations on the conference circuit.

I would like to thank my friends and family for their support; Mum and Dad; Pete, my best friend, for everything; Barney Pell, guru *in absentia*, mainly for letting me beat him at puyo-puyo; the now disbanded Strafillan crowd (Claudio, Jen, Gordon); Tim for lunch and discussions, Sebastien for teaching me how to lose at WarCraft; Robert for distracting me with online shopping; Shibata, Sota, Ichimura for looking after me in ETL. Finally I would like to thank Wakako. She has beaten a path from Scotland to Japan and back again to be with me while I've plodded on with my studies. She deserves a lot more than I can give in return for her love and support.

Abstract

This thesis looks at the issues relating to the development of technology capable of processing tables as they appear in textual documents so that their contents may be accessed and further interpreted by standard information extraction and natural language processing systems. The thesis offers a formal description of the table and the description and evaluation of a system which provides instances of that model for table examples.

There are three parts to the thesis. The first looks at tables in general terms, suggests where their complexities are to be found, and reviews the literature dealing with research into tables in other fields. The second part introduces a layered model of the table and provides some notational equipment for encoding tables in these component layers. The final part discusses the design, implementation and evaluation of a system which produces an instance of the model for the tables found in a document. It also discusses the design and collection of a corpus of tables used for the training and evaluation of the system. The thesis catalogues a large number of phenomena discovered in the corpus collected during the research and provides appropriate terminology.

Figure 1: A tablet of the Fara type (*c.* 2600 BC); a record of numbers of workmen — from [Hoo90].

STOCKMASTER Portfolio Stocks Funds Markets Trade Discuss

Please Support Our Sponsor:

Trading Floor

DATEK ONLINE

$7 Trades ScoTTrade

Name or symbol: IBM for Quote and Chart Go

- Quote & Chart - Detailed Quote - News - Profile & Financials - SEC Filings - Earnings History - Message Board - Research Links -

CBS MarketWatch: NewsWatch | Headlines | Market Data | Home

Page Viewed: Wed May 26 1999 3:51:56 AM EDT

usbancorp Piper Jaffray MULTEX INVESTOR NETWORK

IBM Corp (IBM)

Last	High	Low	Volume	Div	As of
221 3/16	226	221	4,702,500	$0.24	close on May 25
Change	**YearHigh**	**YearLow**	**P/E**	**Yield**	**Exchange**
-2 9/16 ⬇ 1.14%	246.00	106.00	31	0.43%	NYSE

S&P 500 1284.40 -22.25 ⬇ 1.70% DJIA 10531.09 -123.58 ⬇ 1.15% NASDAQ 2380.90 -72.76 ⬇ 2.96%

Chart Options: 6 month 1 year 3 year 5 year

[**LEGEND**] Data delayed 20 minutes. Please read the terms of use.

This research is provided by Multex - you will leave the StockMaster site

Investor Research | Stock Research | Broker Research

News Headlines: by Datek Online

DATEK ONLINE

May 25 12:53 BW IBM IBM Expands Its Computer Telephony Business Solution

Figure 2: A complex document from the web (*c.* 1999 CE), courtesy of StockMaster (www.stockmaster.com).

Contents

List of Figures

Many of the earliest known examples of writing are tables. Archaelogists have found inventory tables that are over 5000 years old. The Babylonians kept multiplication tables and tables of reciprocal values in 2000 B.C. The Greek mathematician, Ptolemy, authored a mathematical table in his *Syntaxis Mathematica* which gives the values of the chords of a circle at intervals of one half degree to a six place approximation.

Since ancient times, tables have continued to be an essential element in writing. Copernicus included a table of sine values in *Concerning the Revolutions of Heavenly Bodies* published in 1534. Logarithmic tables and navigational tables were in extensive use by the early seventeenth century. Important scientific table writers include Kepler, Euler and Gauss. ([Cam89])

Tables are so numerous today, that we even have multi-volume indices and bibliographies of tables [Goe87]. ([Cam89])

[T]able markup contains a great deal of information about what a table looks like ... but very little about how the table relates the entries. ... [This] prevents me from doing automated context-based data retrieval or extraction. ([Tho96])

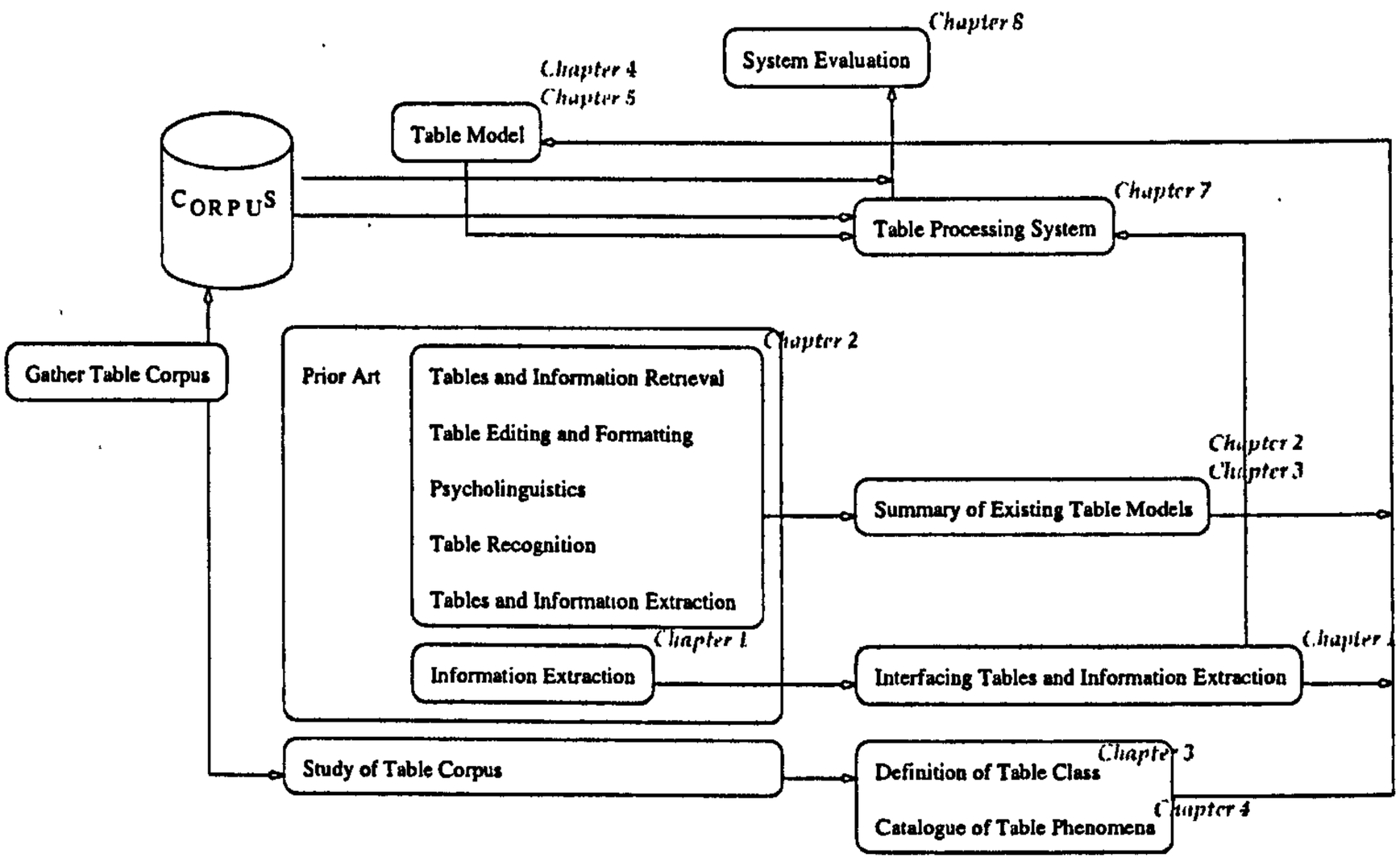
The main difficulty is to separate the essential semantics of a table from its visual layout. At one extreme, a table can be considered merely as a rectangular array of data, and everything else pure presentation. Another, opposite view, says that the whole layout is an important part of the table and cannot be divorved from the data within. A better view will lie between the two. ([Tho93a])

Preface

Organisation of the Thesis

The thesis is organised into three main parts. The first part deals with motivational issues and reviews a number of related research fields which offer some insights into the representational requirements for table capable information systems, the state of the art and the general utility of tables. The second part looks at the phenomena displayed by tables and presents a formal representation of tables. The third part looks at the implementation of a system capable of delivering a description of a table suitable for an information extraction system.

The components of the thesis map onto a goal oriented view of the research as described by the following diagram.



History of Research

The research reported in this thesis has its root in the CISAU system. CISAU was an information extraction and cross-verification system developed by the Language Technology Group at the Human Communication Research Centre in the University of Edinburgh, and BICC, Quantum House. During the course of that research, we realised that many of the documents which the system had to cope with had certain

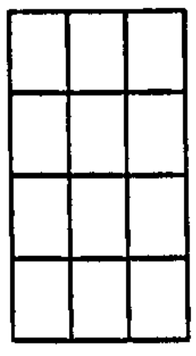
structural aspects which lent meaning to the content. In addition, there were many table and list elements in the documents.

Some amount of research was carried out in the limited time available on the project ([DHQ95]). Additional publications have been produced during the course of this research ([HD97] — some preliminary experiments in extracting structural information based on a simplified model of tables (*cf.* [NLK99]); [Hur99a] — a summary of initial results presented in full in Chapter 8; [Hur99b] — an overview of the model presented in Part II).

Typographic Conventions

Tables used as examples in the thesis are displayed like the following example and referred to like this: **Table (1)**.

(1)



To keep the flow of discussion, a number of examples are included more than once to reduce the need to look back for the relevant example. The majority of examples are from real tables found in documents (see the note on the corpus above). Those which are not are marked with a '!' preceding the reference number next to the table.

In general, terms are introduced in a **bold font**. If they are mentioned before they have been reasonably defined then they are set like this, in a sans serif font. Terms used consequent to their introduction are set in the normal font.

There are three indexes to the thesis. The first is an index of the terms used. The second is an index of authors indicating either where they are explicitly mentioned, or where one of their publications is cited. The third is an index of they computational systems mentioned and discussed in the thesis.

Part I

Tables

The purpose of Part I of this thesis is to introduce and define the key concepts and research areas that form and influence the subject matter of the research. These include the logical document, the table, information extraction *and* research focused on tables.

Through analysis of this related work, an understanding of the different views of tables is acquired which forms the motivation for the model which will be presented in Part II. In addition, consideration is given to the nature of information extraction systems in general and a suitable interface between table processing and information extraction systems is presented.

X

Chapter 1

Tables

The purpose of this chapter is to give an overview of the field of information extraction, the utility of extending the document types input to such systems with tables and a discussion of the nature of the interface between an information extraction system and a table analysis system.

1.1 Introduction and Overview

Traditionally, information extraction (IE) systems have concentrated on specific content domains and specific, generally *pro forma*, document structures.[1,2] Within these constraints, significant progress has been made and systems which produce reasonable results in a particular domain can now be engineered. The Message Understanding Conference (MUC), for example [COO93], is one forum in which much of the standard technology, as well as suitable evaluation techniques, have been established.

[1]By **content domain** we mean a particular field of study or topic or theme which forms the focus of a document. For example, conference proceedings contain papers restricted more or less to one content domain (e.g. computational linguistics for the COLING conference). In this thesis, and elsewhere, 'domain' is an overloaded term and when used without any clarification, the meaning must be taken from context. **Document structure** refers to either the ***logical* structure of the document** or the ***physical* structure of the document** — see Figure 1.1 — and can be represented by a tree whose nodes are **document elements**. ***Pro forma* documents** are documents whose structure follows a certain pattern either prescribed or adopted by convention.

[2]Recently, other media have been exploited for similar goals to those of the IE community, including the extraction of information from speech (e.g. [RR99]). Here we refer only to IE from text.

Additional work, such as [ZM95], though not within the MUC framework, generally processes documents restricted in a similar manner by content and structure.

However, many of the documents which we would like to be processed in some manner by automatic methods, particularly in the technical and business domains, both in print and on the web (see, for example, Figure 2), while constrained to various degrees in terms of content, exhibit great structural variety. To date, little has been done to incorporate this structure into the vocabulary of document types acceptable to information extraction systems. It is neither exploited as a resource indicating relationships between entities in the content domain (for example the placement of noun phrases in titles, section headings, figure captions etc. may be indicative of certain topical qualities and relationships), nor for the information contained in certain non-linear textual document elements, e.g. lists, diagrams, tables, forms etc.

In particular, key results or summaries as well as intermediate data important to the content of the document are often reported in a non-textual manner while being referred to or summarised in part or in whole in the textual elements of the document. [PC97], for example, states that

> [o]ften, the gist of an entire news article or other exposition can be concisely captured in tabular form.

Also, [WBMT19], which looks at the application of text compression to the task of text mining, suggests that

> ..., text could be mined for data in tabular format, allowing databases to be created from formatted tables such as stock-market information on web pages.

Accessing this information requires an understanding of the relationship between the language and structure of the document (phenomena which vary from **meta-text**, which describes how to read the document or how to read diagrams and table entries, to overlapping content where the text summarises, highlights or otherwise reflects the content of complex document elements) and an understanding of the nature of the complex document elements whose content we are interested in exploiting.

The detection and understanding of meta-text in the document has an obvious impact on a system's ability to interpret the object text. For example, stating that

`the values for variable` k `are the minimum values obtained in the experiment`

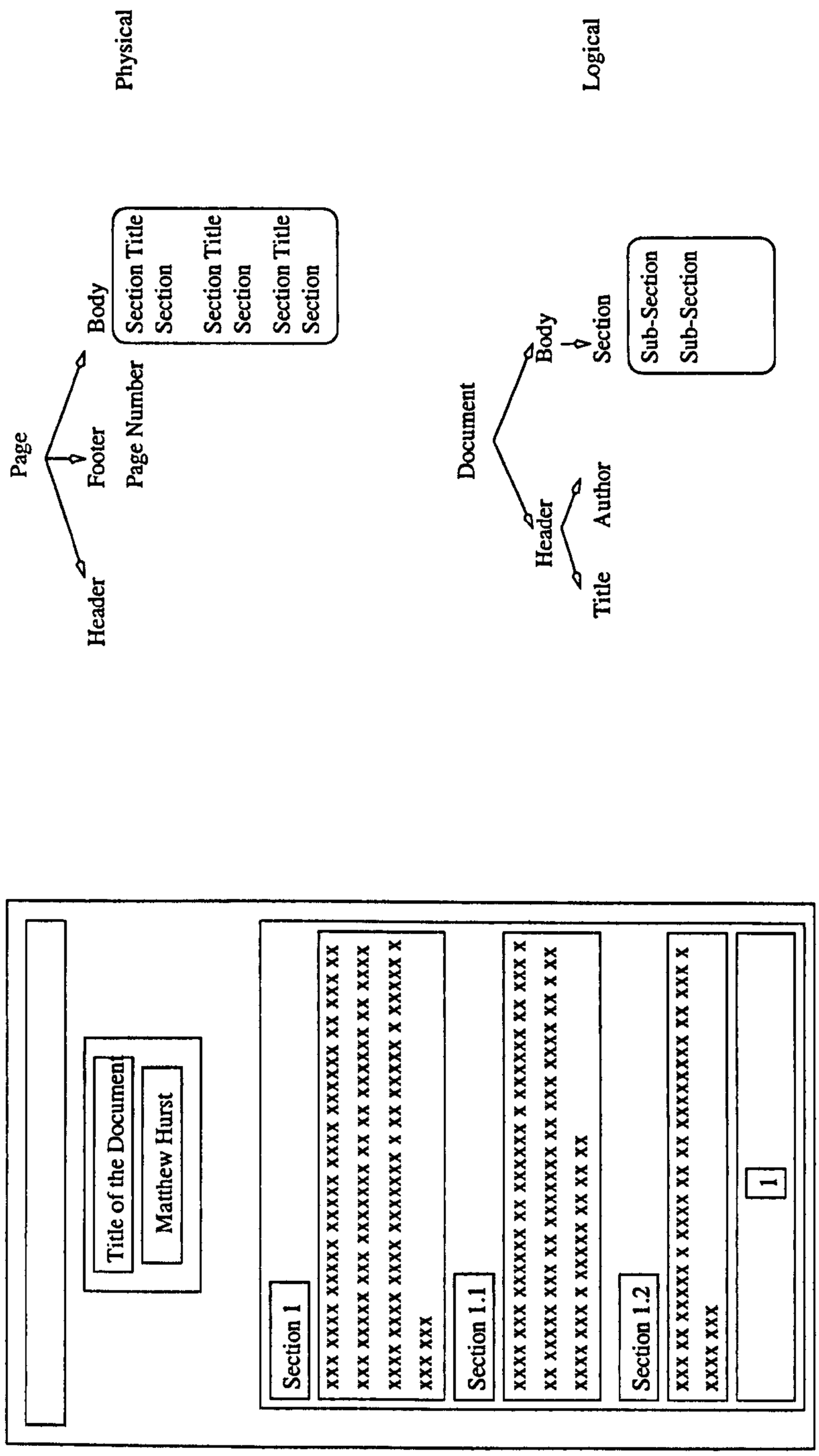


Figure 1.1: Document structure: Physical and Logical

indicates the *meaning* of certain spans of text standing in some relationship to an indication of the value of the variable termed k.

Summary descriptions of a document element can also contain information pertaining to the interpretation of that document element's components. For example, the `table shows that performance improves through time` would indicate that there is some part of the table which indicates performance, and some part which indicates time (e.g. performance might be given in terms of the text `precision and recall`, or `p & r`; time might be indicated via some spatial arrangement, or by some explicit element labeled `date` indicating when a set of observations was made).

After the simple hierarchical divisions representing the gross logical structure of the document, perhaps the simplest step beyond the straightforward prose document element is the tabular document element. However, the table has not received much attention from the information extraction community (except [SBW97] and [LV92]), nor the information retrieval communities (except perhaps [PC97], [KW98]), despite a considerable body of work in the image analysis field ([HD95], [WLS94], [WLS93], [GK95a]), psychological and educational research ([WF70], [WHL84], [GWK93]), and document markup and formatting research ([Tho93a], [Wan96], [Van92])[3]. Possible reasons for this include:

- Lack of current art and model. While the sentence and discourse level models of documents currently used in information extraction systems enjoy a long history there is a relatively paucity of work reporting models of tables or the linguistic aspects of complex document models. Though models of certain 'ontological' components of tables exist (most notably [Wan96]), there is no overall model either associated with a particular high-level task, or a stand alone declarative description.

- No corpora. Current information extraction techniques rely heavily on corpora-based training to parameterise their subprocesses. There is no corpus of tables available for this type of statistical analysis. This problem is in some ways related to the lack of a complex model: though there are many examples of documents marked up with tables, there are no examples of documents marked up with tables including some form of 'result' associated with a table analysis process. It is the existence of such training sets that is important, in general,

[3] For an overview, refer to the literature survey in Chapter 2.

to analyse procedures based on machine learning techniques and to evaluate systems.

- Confusing Markup. There are many table markup systems available[4], however, it is not clear which should be adopted and exactly what it is marking up. For example, the tables section of the HTML definition is an abstraction of the physical layout of the table mixed in with a system which is partly syntactic and partly semantic in nature.

Through the various niches of table-related research there is a lack of evolved or complex representations which are capable of relating high- and low-level aspects of tables. For example, where some research discusses the complexities of detecting and exploiting table line-art (the use of vertical and horizontal lines in the table) in order to segment the areas of a document which might be regarded as cells in a table it will fail to consider the relevance of cell content in distinguishing the purpose or function of those cells. The result is a physical description of the table based on graphical features. If line-art is only partial, or poorly realised, then the resulting ambiguities can never be resolved by such a low-level model (see Section 4.2.2).

However, as this thesis makes clear, the relationships between the components of a complex model may be exploited to improve the performance and the applicability of a table model. The requirement that document understanding systems cannot succeed using purely physical analysis techniques is mentioned in [Niy94], p 100:

> ... spatial information alone provides a significantly large amount of knowledge that enables an effective logical structure derivation. However, a more complete understanding of document structure and content can obviously be obtained by involving text understanding in the logical structure derivation process.

However, no work to date looks at the content of the document as a whole, nor the content of the particular document elements which are being processed to aid the analysis process.

An additional important point is the variety of formalisms and level of formality with which table representations are expressed. In this thesis a **model** is a rigorous

[4]See Appendix C for an overview of these techniques. Markup in general using SGML is discussed in Chapter 6.

characterisation of the components that make up a table and the relations that can hold between them. As indicated above, the table is an elusive and ill-defined document element. A definition will be proposed in Chapter 3 based on the discussion contained in this chapter and the review of other table related research found in Chapter 2.

1.2 Tables in Use

The use of tables varies across domains and genres of documents. Figure 1.2 indicates the number of tables found for particular examples of types of documents and demonstrates this variety. Not surprisingly, works of fiction contain no tables (unless in the exceptional case where a table is used to illustrate some detail or other). Books of a technical nature may contain any number of tables. Newspapers often report financial and other statistical information in tables.

It is interesting to note that for the set of conferences papers examined (COLING, an international conference on Computational Linguistics held biannually), the number of tables per page increases quite steadily over time. This may be a reflection of a number of factors including the increasing ease with which high quality camera ready copy can be produced by the author (in general via the use of systems such as LaTeX and Word) and changes in the nature of the domain (the movement from symbolic to statistical techniques in Natural Language Processing (NLP) as well as the importance of statistical evaluations of systems).

1.3 Information Extraction Systems

As stated, IE systems are used in this thesis to motivate work on the development of table processing technology. To this end, a number of IE systems are examined and a summary presentation of their history and development is given. Although IE systems may be applied to almost any topic, the selection examined here was taken with the MUC conferences as a point of reference. One of the systems is specifically engineered for the MUC task, one is a more general system which was later targeted to MUC and the third is a non MUC system.

Generally, IE is best illustrated through an example. The following text is typical of the input to IE systems of the MUC variety.

Document	Unit	Total Tables	Total Pages
Pride and Prejudice (Jane Austen)		0	
COLING 84	Conference Paper	49	562
COLING 86	Conference Paper	47	675
IJCAI 93 Volume 1	Conference Paper	59	836
IJCAI 93 Volume 2	Conference Paper	105	872
COLING 94 Volume 1	Conference Paper	104	639
COLING 94 Volume 2	Conference Paper	123	660
COLING-ACL 98 Volume 1	Conference Paper	235	741
COLING-ACL 98 Volume 2	Conference Paper	245	766
Understanding Japanese Information Processing		387	435
Sunday Times		22	134
WSJ 1987-1992		6509	15MB
Highway Code		3	100

Figure 1.2: The frequency of tables as they occur in documents from different domains and different genres. (WSJ data quoted from [PC97].)

```
<DOC>
<DOCID> wsj93_050.0203 </DOCID>
<DOCNO> 930219-0013. </DOCNO>
<HL>    Marketing Brief:
@  Noted.... </HL>
<DD> 02/19/93 </DD>
<SO> WALL STREET JOURNAL (J), PAGE B5 </SO>
<CO>    NYTA </CO>
<IN> MEDIA (MED), PUBLISHING (PUB) </IN>
<TXT>
<p>
New York Times Co. named Russell T. Lewis, 45, president and
general manager of its flagship New York Times newspaper,
responsible for all business-side activities. He was executive
vice president and deputy general manager. He succeeds Lance
```

```
R. Primis, who in September was named president and chief
operating officer of the parent.
</p>
</TXT>
</DOC>
```

Note the inline SGML tags, the header information describing the document and the depth of the markup: to the paragraph level. Output would be of the form below which describes firstly the entities which were discovered, and then some of the relationships between those entities.

```
<ORGANIZATION-1>
    NAME            : "New York Times Co."
<ORGANIZATION-2>
    NAME            : "New York Times"

<PERSON-1>
    NAME            : "Russell T. Lewis"
<PERSON-2>
    NAME            : "Lance R. Primis"

<SUCCESSION-1>
    ORGANIZATION : <ORGANIZATION-2>
    POST         : "president"
    WHO_IS_IN    : <PERSON-1>
    WHO_IS_OUT   : <PERSON-2>

<SUCCESSION-2>
    ORGANIZATION : <ORGANIZATION-2>
    POST         : "general manager"
    WHO_IS_IN    : <PERSON-1>
    WHO_IS_OUT   : <PERSON-2>

<SUCCESSION-3>
    ORGANIZATION : <ORGANIZATION-2>
```

```
    POST          : "executive vice president"
    WHO_IS_IN     :
    WHO_IS_OUT    : <PERSON-1>

<SUCCESSION-4>
    ORGANIZATION : <ORGANIZATION-2>
    POST          : "deputy general manager"
    WHO_IS_IN     :
    WHO_IS_OUT    : <PERSON-1>

<SUCCESSION-5>
    ORGANIZATION : <ORGANIZATION-1>
    POST          : "president"
    WHO_IS_IN     : <PERSON-2>
    WHO_IS_OUT    :

<SUCCESSION-6>
    ORGANIZATION : <ORGANIZATION-1>
    POST          : "chief operating officer"
    WHO_IS_IN     : <PERSON-2>
    WHO_IS_OUT    :
```

1.3.1 A Brief History of Information Extraction

This brief overview of the history of the IE task has been produced from inspection of a number of publications ([Col96], [Wil97], [HKS96], [MUC95], [CHL93], [GW98], [GS96], [LS91] and [CGW95]). The table in Figure 1.3 traces some of the roots of IE as well as the history of the systems discussed in later sections. The development of the concept of information extraction passes through earlier work on message understanding. **Message understanding** research has a more general goal, that of understanding the entire document, not just targeted fragments. However this process is often *embedded* in the architecture of current IE systems, as discussed below.

In general, IE systems were targeted and produced various forms of output. The event of the MUC conferences kept the targeted domains but rationalised the output

Date	Event/System/Researcher	Comment
1970	Sager's system	patient discharge summaries to database
1979 (and 1982)	deJong and FRUMP	script based system, newswires to event descriptions.
1980	DaSilva and Dwiggins	satellite flight information.
1981	Cowie	field guide descriptions of plants and animals to canonical structures (fixed record).
1982	DIALOGIC ([GHH+82])	SRI International's parser later to be used in TACITUS
1983	Zarri	activities of french historical figures to relationships and meetings between them.
1985	TACITUS system	SRI International starts development
1987	MUCK1	short naval messages
1989	MUCK2	short naval messages
1991	TACITUS system	SRI International's entry to MUC-3
1991	MUC-3	newspaper and newswire on terrorism, translated from spanish.
1992	FASTUS	SRI's new approach.
1992	MUC-4	newspaper and newswire on terrorism, translated from spanish.
1992	POETIC	Sussex University's traffic incident report message understanding system. Later to be modified for Sussex's MUC-5 system
1993	Sussex MUC-5 system	experience of which passed on to LaSIE
1993	Diderot MUC-5 system ([CGJ+93])	experience of which passed on to LaSIE
1993	MUC-5	
1995	LaSIE	developed from experience at Sussex and New Mexico.
1995	MUC-6	
1995	MENELAS	patient discharge message understanding system developed at the University of Edinburgh, Language Technology Group

Figure 1.3: Summary of some IE milestones

format, and later the system architecture, through a number of sub-tasks (mini-MUCs). A number of researchers have taken their basic systems and applied them to different domains in an effort to investigate the cost of retargeting and as a means to evaluating the generality and robustness of their systems.[5]

None of the systems (except the published claim that SRI's FASTUS system was reworked into the Warbreaker Message Handler System which included a table processing module) have been transferred across document types of any significant variety in terms of the complex layout of the document type (though applications have been ported to new content domains, e.g. [HDG00]).

1.3.2 System 1: MENELAS

MENELAS 'An Access System for Medical Records using Natural Language' ([ZM95]) was a large European project which looked at the problem of understanding patient discharge summaries (PDS) with the goal of supplying support for those dealing with medical information systems. Due to its European nature, a number of sub-systems were developed in a number of languages, but only the English system will be discussed here.

MENELAS is actually a message understanding system, not an IE system, however it is still of interest due to the scale of the project (which introduced a number of different solutions to component problems) and to the fact that an 'off the shelf' grammar was used, the development of which indicates the utility of such resources and the problems associated with tuning it to a certain domain.[6]

A large-coverage grammar was used together with a number of pre-existing NLP sub-systems in the form of the Alvey Natural Language Tools. However, in order to improve the scope of the grammar, the documents were preprocessed to identify certain semantic units which could be dealt with in an unanalysed syntactic format. These included names (hospitals, drugs, doctors and so on — similar to the named entity sub-task in the MUC conference) and dates. Additionally, some processing was done to deal with typographic errors.

---

[5]Systems have been implemented and reimplemented in a number of languages including Prolog (Sussex, LaSIE), LISP (CIRCUS, [LMS+93]) and C++ (FASTUS, [HAB+]).

[6]The use of 'of the shelf' components is very much behind the development of the system presented later in this thesis. In doing this, a form of evaluation is being carried out on the maturity of the NLP field and the quality of the resources that have resulted.

The documents input to the system had a particular structure consisting broadly of a header and a body. The header is *pro forma* and its elements (names, addresses, dates and locations) can be extracted based purely on the physical layout of the text. Interestingly, the language of the PDSs was in a 'free-flowing' style, and didn't contain the telegraphese[7] found in some IE document types.

The large-coverage grammar supplied semantic descriptions for the leaves of the syntactic tree produced by the parsing process. These semantic descriptions were combined in the normal manner (as for lambda calculus): semantic representations for sub-parts are combined up the syntactic structure until the meaning of the sentence is represented by a logical sentence which has been composed of the original semantic constituents. This semantic analysis then requires to be mapped to the conceptual representation system used to model the world as described by the PDSs. This conceptual graph formalism is the representation which holds the interpretation of the document. In contrast with the specific templates required by the IE task, MENELAS tries to describe as much of the document as it can, and doesn't focus only on particular activities and agents. However, the restricted nature of the domain results in a complexity broadly similar in scope to that of the prescribed IE task.

1.3.3 System 2: LaSIE

Sheffield's LaSIE system [GWH+95] adopts a reasonably traditional approach to NLP in general, though it relies on a number of resources which are automatically derived from corpora and are statistically motivated: a lexicon is replaced by statistically derived tags and a morphological analysis, the grammar is derived from a corpus. Another one of its advertised features is the manner in which resources and results are used across various different MUC tasks (e.g. coreference information is used in the identification of named entities). In addition to the standard suite of MUC tasks, LaSIE, as a bonus, can also generate brief natural language summaries of the extracted events. Essentially, this is offered as evidence of the system's generality (*cf.* FASTUS, Section 1.3.4).

LaSIE uses a chart to structure initial lexical and syntactic information (lexical and some semantic items are used to seed the chart) and a bottom up parser is

[7] Telegraphese is the term used to describe language which is clipped and reduced to impart information with the minimum of redundant linguistic material, as found in telegraph messages.

used. A 'best parse' is selected and a semantic analysis is built up from this parse. Discourse interpretation then converts the semantic representation in to a model containing extra information such as ontological classes and their properties. Coreference resolution is carried out in a procedural manner which is also used to equate variations of names (e.g. `Ford Motor Co.` and `Ford` are identified as referring to the same entity).

The resulting discourse model is then scanned for circumstances which fit the predefined templates of the MUC task, and the results are generated.

1.3.4 System 3: FASTUS

SRI's FASTUS [HAB+] takes a novel approach to the problem of IE. As a reaction to the fragile and computationally expensive systems which they had derived from their more traditional computational linguistics systems research (TACITUS [Hob91]), SRI decided on a different engineering solution. Trading a certain amount of generality and linguistic 'hygiene' for speed and clarity, they built a system which uses finite state automata (rather than a context free language formalism) and which is integrated through a cascading data flow. The raw document is input and a series of modules produce output representing a certain stage of analysis, which is then fed in to the next module.

The pragmatic approach to the problem produces a very much goal-oriented philosophy as is summed up in the following:

> The task of the system is to build templates or database entries with information about who did what to whom, when and where. [HAB+]

This, in some ways, doesn't apply itself well to the underlying goal of the MUC Conference which is to advance the fields of IE and computational linguistics in general. Although FASTUS may produce a suitable solution to a very specific problem it does not say much about language. Its ability to be generalised over document types, for example, is not clear.

The modules which the system incorporates are as follows:

1. Names and other fixed form expressions are recognised.
2. Basic sentential syntactic components are found: noun groups, verb groups and prepositions (plus some others).

3. 'Certain' complex noun groups and verb groups are constructed.

4. Using patterns, interesting event are identified and "event structures" are constructed.

5. Event structures which describe the same event are recognised and merged, and a data base is generated.

Speed was one of the major concerns when the system was conceptualised, and in that at least it performs far better than the previous SRI system : 36 hours for 100 articles versus 12 mins.

In summary, FASTUS attempts not to provide a linguistic interpretation of text, but to discover certain phrases via shallow processes and then to find the relationships between these phrases in order to place them in a database. It does demonstrate that a lot can be done via shallow processing and a very pragmatic approach. An interesting point made in [HAB+] is the following:

> We currently have a version of the system, a component in the Warbreaker Message Handler System, for handling military messages about time-critical targets, which has a preliminary stage of processing that identifies the free and formatted portions of the messages, breaks the free text into sentences, and identifies tables, outlines, and lists. The table processing is describe in [TAH+] (to appear).

However, to date, [TAH+] is *still* 'to appear'.

1.3.5 A Characterisation of IE Systems

In [Hob93] Hobbs provides a characterisation of an information extraction system.

> An information extraction system is a cascade of transducers or modules that at each step add structure and often lose information, hopefully irrelevant, by applying rules that are acquired manually and/or automatically.

Although the systems described above take two different approaches to providing the same output, they all still fit this basic description. However, the LaSIE system allows the flow of information in both directions to a certain extent, clearly a novel

development with respect to the systems Hobbs had in mind when writing the above. In fact, the notion of levels of information and the direction of flow is one which will be used in the development and discussion of the model of tables at the centre of this thesis.

In addition to characterising the features of IE systems it is relevant to look at the tasks they are set to and the nature of the documents which are relevant to that task. One way in which a task can be delimited is by the domain in which the system is to process texts. It is unheard of for any system not to be restricted in this manner. The retargeting of a system to a new domain is frequently used as an indication of the generality of the system and the tools which go to make up its resources (grammars, taggers, lexica, ontological knowledge etc.). The second dimension in which restriction may occur, and one of which little mention is ever given, is the type of document. All of the systems above are looking for *specific* types of information in *specific* types of documents (newswires, patient discharge forms, news bulletin transcripts etc.). Interestingly, according to personal communications with IE system developers, when faced with a recognisable table, current systems simply remove that portion of the document. This, contrary to Hobbs' definition, loses information which is *not* irrelevant.

In [CL96], a number of hypothetical tasks are presented for the application of IE technology. It is an interesting experiment to search for documents (on the web) in the domains of those tasks and look at the type of documents which are found.

One of the hypothetical domains is the tracking of forestry data for different countries in the import and export market. Although the evidence is anecdotal, it is clear that the documents retrieved in this search are complex in that they include more than simple section headings and paragraphs of text. Much information was presented in tabular form (as might be expected for financial information). Very few of the documents would fit the type traditionally used by the MUC task (i.e. simple newswires).

1.4 Towards A Suitable Interface Between Information Extraction and Table Analysis

As the aim of this research is to provide and quantify *domain independent* technology (*i.e.* representational systems which can be described and procedural systems which

can be evaluated) for enabling IE systems to take documents with tables as input, it is important to clearly state the product of the table processing system, its rationale and its proposed relationships with a generic information extraction system. To these ends, the desired output of a table processing system is motivated and discussed. Understanding something about the possible interface between an IE system and a table processing system provides insights into both details of the model and areas of ambiguity.

To what level of representation can the domain independent processing of tables be taken with respect to a generic information extraction system?

We want to establish the level that the IE system will involve its NLP processes on the models of the tables built up during the document or table analysis stages. This is the point in the processing of the document where all the resources of the full IE system come to bear on the representations built by the table processing sub-system. The resources that the IE system has to offer will include processes to provide a semantic analysis to phrasal units which are relevant to the domain, processes which can describe the relationships between recognised entities in the domain, processes which can resolve two descriptions of the same object to a unique referent and so on.[8]

Given the logical/linguistic nature of IE systems, we are ultimately concerned with creating a map of possible and typical relationships between identifiable (linguistic) components of a table and a description of how such an analysis might fit into a typical IE system. The development of the model which contains and supports algorithms which identify these relationships forms the majority of Part II of this thesis. Discussion of the manner in which such an analysis is included in an IE system requires that we consider firstly how IE systems work in general, the differences between the tabular presentation of information and the presentation of the same or similar information in a prose form, the issue of whether it is appropriate to try and apply the IE analysis model to tabular information at all, and - if the answer is yes - how we might do this.

Of course, it is possible that in some cases such an interpretation — i.e. one which discovers which linguistic components 'interact', and then combines them — is *nearly*

[8]It should be noted that there is also an attractive reversibility to this discussion whereby knowledge about a domain can be discovered via assumptions made about the table analysed in a domain independent manner.

all that is needed. In the following, a system might assume that relationships hold between 'column and row headers' and the cells aligned with them, and that the nature of those relationships can be determined from an analysis of the content of the cells.

(!1.1)

	Age
Matthew	28 years
Peter	29 years

In **Table (1.1)**, if we take the 'identifiable' linguistic components to be {age, matthew, peter, 28, 29, years} then we might end up with an output similar to `_matthew.age.28.year` (equivalent to 'the' `age` 'of' `matthew` 'is' `28 years`) which might in turn be transformed into `age(matthew, 28)`. In other cases, the relationships between even simple noun groups can be more complex. For example **Table (1.2)**.

(1.2)

Type	Doctor
EU Student & PE Staff Member & Non-Member	$5.00
Sports Bursar (ACE)	$5.00
EU Staff Member	$13.00
Other Member	$14.00

In **Table (1.2)**, the label `Doctor` indicates the doctor's fee and not the name or other identifier of a particular doctor. Additionally, in some cases, we find complex linguistic relationships akin to ellipsis between cell contents. In **Table (1.3)**, the cell containing the text `No Of Women` requires satisfaction with missing linguistic material. This material can be taken from the adjoining cell: `stopped for importation`.

(1.3)

Substance	No Of People Stopped For Importation	No Of Women
Synthetic drugs (Ecstasy/amphetamines/LSD)	248	28
Herbal cannabis	905	135
Cannabis resin	1190	134
Cocaine	311	102
Heroin	124	20

In the first two examples above (**Table (1.1)**, **Table (1.2)**) it can still be argued that a system set to analyse the relationships between the noun groups in the cells can rely on certain closed classes of semantic types to imply the relationships. For example, it is not unreasonable to suggest that all strings indicating a price might be recognised by automatic means and consequently, the relationship 'price_of' between the interpretation of the 'labeling cell' and the price be implied.[9] However, there are further simple cases where more types of resources are required. Consider **Table (1.4)**.

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

In **Table (1.4)**, the relationship between `Houston` and `CITY` might be described as `type_of` or `instance_of`. It would be unreasonable to assume that, for every such case of this relationship, we might be able to recognise the instances using some *domain independent* process. One course that can be taken is to use general knowledge sources such as WordNet ([Fel99]). However, there is still no guarantee that such relationships will always be found, or that entirely new ones are not defined in the body of the document. Consequently, we can say that:

The domain independent processing of tables can be taken as far as identifying the linguistic objects (e.g. 'CITY', 'Houston') which stand in some relation to each other (e.g., 'instance_of', 'price_of', etc.), but can not necessarily identify what those relationships might be.

This statement can be reversed to suggest something about the nature of the

[9] In fact, this is not the case. The relationship between a quantity of money expressed in terms of a specific currency might indicate the price of the 'label' which dominates those values, however it could also indicate the value of a particular quantity, for example the amount that a film took on a particular date (e.g. [4.46]). Consequently, although such heuristics regarding the nature of the semantic relationship between the contents of cells based on the semantic type of those cells' contents are useful, they are ambiguous and require further knowledge and processing to accept or reject. This indicates the limit to the independence of a table processing system for IE.

table in general: a table is a device used to present information to the reader by organising some set of meaningful elements on the page so that the relationships between those elements, and the manner in which combinations of the elements interact, is demonstrated to the reader. This definition is not restrictive enough, but it does go some way to understanding the general nature of the table. What this definition hides is the great amount of ambiguity inherent in the tabular presentation of information and the consequences for table processing systems of this ambiguity. It further underlines the fact that some of the relationships holding between elements of the table are 'known' pieces of information which are being exploited by the author as his assumptions about the reader's knowledge allow the reader to index the information in the table. Other relationships are 'new' information which the table is presenting.

What becomes interesting once the level of the domain independence of the problem is established, is identifying the *types of relationships* which exist between the linguistic units of cells and extending the domain independent power of the analytical system to identify when a reasonable guess can be made (e.g. based on the recognition of a closed class of semantic textual units such as dates, units of measure, prices etc.), when the text of the document containing the table might be examined, again, using shallow processes and a model of the manner in which the content of tables is discussed in text and what implication this model might have regarding the relationships between the linguistic elements[10], and when world or domain knowledge is required to fill in the gaps.

In summary, we can identify the following types of analysis:

1. closed classes of semantic units implying relationships.

2. exploiting a model of discourse, meta-text and shallow processing to reveal relationships.

[10]Of the various kinds of meta-textual material found relating to tables (which will be discussed in greater detail later), a key example is that type which tells the reader *how* to understand the table: the decoding procedure. This material might mention something about the content of the cells (`the values in the first column are the maximum found when ...`) or it might be about the relationships between cell elements (`'p' indicates a positive increase, whereas 'n' indicates a negative one`). In addition to the above, we must also deal with reference in the table to more explicitly defined terms in the document. This is essentially the same as the type of coreference which must be carried out between entities discovered in sentential IE documents (see Section 1.3.3).

3. employing world knowledge (e.g. the names of cities, stars, personalities) to group siblings and discover relationships.

4. employing domain knowledge to discover relationships.

Integrating table processing with information extraction requires that the table processing system delivers a description of the table in which the navigation of cells is represented and that the 'location' of relationships between linguistic fragments is identified. Beyond this, we can develop semantic analysis processes which work in the above four areas.

1.5 Chapter Summary

This chapter has motivated the extension of generic information extraction systems by the inclusion of complex document elements (specifically tables) into the range of document types which the systems can handle. It has stated that such an extension is not only necessary to deal with a more realistic domain of input but will also have a number of beneficial effects including knowledge extraction.

The chapter then outlined the field of information extraction, discussing a number of example systems, and concluded with a discussion of the manner in which domain independent processing of tables might be integrated with information extraction systems.

The basic task of the thesis is to develop a model of tables consistent with the phenomena observed in as large a collection of examples as possible and which can be used to extract information from tables appearing in complete documents.

Chapter 2

An Overview of Tables-Related Research

This chapter introduces and summarises fields of research which are concerned with the study of tables in some form. The main aim here is to introduce different views of the table and its features important to different types of research. In summarising the work, a catalogue of table models is built up from which the requirements of an overall model may be taken.

2.1 Table Recognition and Segmentation and Table-Form Analysis

The recognition of tables in documents is either a research goal in its own right, or the first step in an integrated system such as those created for IE or IR. According to [GK95b],

> The recognition problem is to locate and characterise the cells of a table in a two-dimensional black and white document image.

Perhaps the main dimension along which research in this field varies is the class of principal features which are used and relied upon by the various segmentation algorithms. Some systems ([GK95b]) use line-art[1] as the main visual key for recognising

[1] **Line-art** is the term used to describe the arrangement of lines enclosing or delimiting the content of cells. Section 4.2.2 discusses the issues relating to line-art in more detail.

tables. Others use white space ([RS94]), or low level image semantics (e.g. box driven reasoning in which text and lines are represented in terms of their areas and extent — a tiling of the document with boxes, and the hierarchical relationships between the boxes according to their type (text block, image, heading block, etc.) ([HD95])). Finally, there are those which use some notion of contiguous textual features where areas of the document are classified according to the 'density' of the characters in certain regions of the page ([KD98]).

Another key classification is the use of a model versus a more data driven approach. In some cases, the use of a model *of a specific table or form* is appropriate due to the specific target of the application ([SBW97]). In other cases, a model of tables is used (e.g. [WLS93], [WLS94] which bears many similarities to XY trees (see Chapter 3) as used by, e.g., Green ([GK95b])).

The component of the TINTIN IR system ([PC97]) which locates tables in documents uses white space as its main feature. Essentially, it extracts information regarding certain types of white space context (within a line, with respect to alignment with white space in neighbouring lines) and uses these to guess the function of the line: plain text in a paragraph, a table row and so on. This can be contrasted with systems like T-Rex ([KD98]) which use features based on the relationships between textual components (space bounded 'words'). For TINTIN, a table is made up of captions and table elements. The captions include what would normally be called the title (if positioned at the top of the table) and the head. The table elements are the rows in the table below the head (thus the system makes no special allowance for the stub (the left most column or complex of cells)).

Table recognition and analysis, in general, seems to produce well defined approaches within research groups which are in some ways reluctant to interact. Chandran and Kasturi ([CK93]), for example, recognise that using simple line-art techniques is not enough due to the presence of many tables with little, no or inconsistent line-art. However, in stating that '*tables must be treated as graphics in order to extract the structural information and the contents of the table should be extracted using character recognition methods made accessible through this structure*' they fail to recognise the requirement that all levels of the table need to be integrated if a truly high performance system is to be developed (*cf.* [KD98] which doesn't exploit line art).

In summary, we can view this field has having a number of specific goals.

1. Recognise where the tables are in the document.

2. Recognise the delineation of cells in the table and segmenting the cells.

3. Homogenise the cells in a table.

All but the last are generally discussed in the literature. **Homogenising** or quantising the cells once recognised is the process of recognising the aggregation of cells according to a simple grid based view of the physical table. In other words, although it is possible (as [KD98] demonstrates) to recognise the boundaries of cells, it is another task to correctly identify the columns and rows of cells. A simple example of this problem is demonstrated in **Table (2.1)** and **Table (2.2)**. If no line-art were provided in **Table (2.1)** then there is no cue to recognising the span of the cell Dam.

(2.1)

	Dam		
Sire	Black	White	Grey
Black			
White			
Grey			

(2.2)

		Dam	
Sire	Black	White	Grey
Black			
White			
Grey			

Another general problem is aligning cells. Providing the analysis in **Table (2.3)** fails to indicate the column and row groupings required. This is an illustration of the cell identification task which equates the extent of the text with the extent of the cell; clearly not an ideal assumption as the top-most cell Dam actually extends to the left and to the right of the text centred within it. In addition locating cells does not tell us anything about the physical relationships between cells in terms of some form of quantized grid.

(2.3)

		Dam	
Sire	Black	White	Grey
Black			
White			
Grey			

2.2 Editing and Formatting

Table editing and formatting deals with the problem of supporting authors in the task of creating tables, editing them once they exist and rendering those tables for inclusion in a document. The issues that must be dealt with range from the problem of presentation — formatting the table from some level of representation to the final document image — to capturing the underlying relationships between

basic table 'elements'. An editing system must decide at what level the table will be represented (ranging from the abstract to the physical).

Perhaps the earliest book (according to Beach) on using a computer to typeset tables is [CB62]. Wang also reports the Improv system as being among the first to abstract the logical structure from the physical structure of the tables in the editing process.

TABLE ([BEF84]) is a 'what-you-see-is-what-you-get' editor for tables. One of the prime objectives of this work was to provide an editing environment with polymorphic operators. For example, deleting a character or word in a sentence versus deleting a table cell, column, row etc in a table. The operation is the same but the objects that the operation is working with are of different types. The basic presentation of the table being edited is a matrix or grid structure. Cursor movement is described in terms of an extended cursor which moves between logical objects and uses graphical cues to indicate at which level of granularity an object is being edited.

Although the system offers a number of editing features which are capable of rapidly constructing a table, the underlying model is a lot closer to the physical representation than any abstract notion of the content and organisation of the table. A simple logical representation of the structural table requires at least two hierarchical components (vertical and horizontal). The TABLE system as reported chose between either a grid structure or a single hierarchical structure, opting for the grid.

Beach ([Bea86]) provides a survey of the issues associated with producing a table from a typographic perspective. He describes the table (formatting) problem as being a halfway house between the larger problem of dealing with complete documents, and the smaller scale problem of formatting, for example, mathematical objects.

The publication raises a number of key points regarding the problems which tables bring to the author and publisher. Of particular interest is the recognition of the importance of aligning information in two directions at the same time as

> [i]t is very important to maintain control over placement because the organisation of information in tables is part of the message. Juxtaposition and other spatial relationships within tables have an important impact on the way in which tables convey information.

Improv ([Cor91]), according to Wang ([Wan96] page 16), may well be the first system manipulating tables which separated the logical and the physical structure. A

category in Improv is essentially a row or column — an index to a cell; which requires two categories for complete specification. Categories are linked to the physical table via a link to a 'category tile'. Moving the tile would shift the column, row or both, of a category.

In [Wan96], Wang offers perhaps the most detailed model to date of a table. She introduces a mathematical characterisation of the categories in a table, and then provides a series of operators which can transform a table in certain ways. For example, categories may be inserted and deleted, duplicated, combined, split and so on. Each operation is defined with a description of its effect on the abstract table.

What is missing from the model presented is an account of the semantic relationships between the category components and the constraints and implications these might have on the manipulation of the table model.

Wang's thesis offers the following *desiderata* for a model of tables.

1. The model should capture a wide range of tables.

2. The model should be independent of the presentational form of the table.

3. A well defined mathematical representation should be used.

The formatting and editing system that is implemented from the abstract model of tables (XTABLE) is described as requiring a number of style rules.

1. Presentation-oriented style rules.

2. Content-oriented style rules.

3. Layout-oriented style rules.

Wang also presents the table formatting problem and offers a computational characterisation of it, demonstrating that it is NP-Complete (*cf.* [Bea85]).

Lefrere's comprehensive assessment and feasibility report for the Open University ([Lef89]) states a number of specific aims, among which is the implementation of a program which will support the creation and editing of tables and which will *provide advice* on the table appearance. The advice, which is an interactive component of the creation and editing process, is capable of transforming tables from one 'type' to another.

Lefrere describes ("from the literature") a number of different types of tables (page 9).

- Analytic, or reference tables.

> Analytic ... tables ... have a title and a reference number, they may have many rows and columns; they are usually self-contained, *i.e.* can be interpreted without referring to the rest of the document and can appear before or after the paragraphs of main text which refer to the table; they have structured headings; they use rules; they may have the same left and right margins as paragraphs of main text and any footnotes to the table appear immediately below the table.

- Archival, raw data, appendix or record tables which share all the features of analytic tables but are, presumably, located in a different part of the document, or in different document types altogether.

- Informal, summary, integral or in-text tables.

> Informal tables usually contain fewer significant digits than other tables, for their purpose is to facilitate comparisons (eg by minimising memory load) and to highlight and clarify any patterns, regularities or exceptions.
>
> ⋮
>
> [A] typical informal table has no title or reference number, it has only 1-3 columns and 2-3 rows; it is not self-contained, *i.e.* it cannot be understood easily without referring to the surrounding text, if it is moved out of sequence, relative to the paragraphs of main text which surround it; it may have no explicit headings; it makes no use or rules (except above and below any totals); it is usually indented; and any footnotes to the table appear at the bottom of the page.

Lefrere ([Lef89], 11) also provides a quote from [Woo68] which gives some insight into the processes governing the creation of tables.

> Authors have "private" and "public" purposes in constructing tables. As [Woo68] page 115 points out:

> The private purposes are for clarifying the author's own thinking, and the kinds of table ... that this activity leads to have usually been roughed out before the author has taken any steps at all towards writing the article. [This is in contrast to] ... public purposes - communication of information [- in which] ... the data must be shown meaningfully. Tables ... are supposed to accomplish something: to reveal comparisons or changes and, if possible, to indicate why they are significant.

Table editing and formatting goes some way to representing the logical table (especially Wang). However, none of the literature discusses the nature of the interaction between the logical (or abstract) table, and the physical table, though something is said about the presentation of the table in terms of stylistic rules. An examination of the physical phenomena encountered in tables and the relationships they have with the logical table is presented in Chapter 4 and demonstrates the requirement for understanding why certain arrangements of cells occur, what ambiguities they generate and their relationships with the logical or abstract table. This examination also collects and introduces terminology for describing these phenomena.

2.3 Psycholinguistics

The research into the design, integration and use of tables in terms of the human processes involved we term psycholinguistic research. The main experimental focus of this type of work is the manner in which tables are read, and the effect of certain organisational principles on the speed of locating information in the table.

Although no explicit model of tables is proposed in publications reporting this research, a number of terms are introduced which might be conveniently adopted. Firstly, Wright mentions explicit tables and implicit tables in [WF72]. An **explicit table** contains all the required data points, whereas an **implicit table** is open ended requiring the reader to fill in values according to patterns established in the table. Additionally, Wright introduces list tables and matrix tables in [Wri68]. The matrix table presents information using a head and stub with a number of columns of data cells in the cetral area (which we term the matrix). [Wri82] mentions that matrix tables impose a greater cognitive load on the reader, requiring the memory of one selection while deciding on another, hence the suggestion that redundancy in

tables is not inherently a bad thing. Cameron discusses categories in [GBB91b] (p. 304), which appear to be similar to the '*decision structure selected by the designers*' mentioned by Wright *et al* in [Wri82], and indicates that category selection is one of the important steps in understanding the organisational principles in tables.

> In the proposed model of search, a person inspects a document selectively. That is, the searcher identifies categories that are relevant to the question the person is trying to answer, processes this information deeply, and ignores other categories of information within the document. Most documents contain markers such as row or column headings, labels, and special typography to help the user identify the critical categories of information and thus search selectively.

However, [GBB91b] fails to define categories formally and we can only rely on intuition to provide a simple interpretation. [GBB91b], p 322 also describes the table in terms which suggest a navigable aspect of the table.

> To be selective in inspection of a document, the person must grasp the organisational structure of the informational display and know how to enter this structure.

Two models of the process of table understanding are reported. The first is in [Wri82] and the second is found in [GBB91b]. Wright suggests that humans have to carry out at least the following tasks.

1. Grasp the logical principles on which the information has been organised.
2. Find the required information within the table.
3. Interpret the information once it has been found.

This is similar in style to Guthrie's proposal.

> The model proposes that, to locate specific information in written document such as tables or schedules, the searcher engages in the following processes:
>
> 1. Goal formation — A specific objective is formulated.

2. Category Selection — A category of information from the document [table/schedules/directories etc.] is identified and selected for processing.

3. Extraction of information — Critical details within the selected category are identified and stored in memory.

4. Sequencing — The searcher repeats the above three operations until the full requirements of the goal are met.

The results of experiments on the relationships between the organisation of the table's elements and the speed with which information is accessed is generally that which would be expected.

[GBB91b] p. 323 describes the difference between processing tables and processing prose.

> The major cognitive distinction between document search and prose comprehension is likely to be the process of category selection, which is the basis of selective inspection.

The psycholinguistic research underlines the role of the 'category' in the table, linking this with the model presented by Wang, though it doesn't provide a satisfactory definition.

2.4 Information Retrieval

Possibly unique to the cross over of table research and information retrieval is the TINTIN system ([PC97]). The aim of this research is to exploit the relationship between the structural phenomena (the table), its contents and the content of the query. There are two parts to the research. The first is the identification of the tables in the (unmarked up) document. The second is the goal of creating a system which allows the user to formulate queries sensitive to the particular model of the table they employ.

The model used has two components based on general semantic elements of the table: captions (also known as the head of the table) and table lines. A heuristic approach is used (*c.f.* [KD98]) to recognise the tables in the document. Indexing

information for the retrieval process is extracted from the caption and table line segments of the table.

Currently there is no report of the obvious extensions to this research (mentioned in the conclusions of the paper) — the identification of the functional areas of the table and the requirements on the query processing system to identify terms desired in the data or terms desired in the index structure of the table. Significant as these enhancements would be, the proposal for the apparent functional analysis of the table relies on a template approach to the identification of the appropriate areas which views the table as a series of uniform labeled columns.

> ... slicing the table into columns and treating each column as a document. The column header and body content occurring together indicates more specificity and could be a source of multiple evidence for the corresponding table. For example, if the query is "China Exports Slippers" and we have a table with "China" and "slippers" occurring together in a column, this should get more weight than the case where "Romania" and "slippers" occur in one column and "China" occurs in another one.

In [KW98], Kornfeld and Wattecamps talk in very general terms about the SEC[2] filings domain — documents descibing the financial details of commercial activity. Unfortunately, as the system they hint at is a product not much is given away about what it does or how it does it. However, the paper does serve to indicate a very suitable application for a table-capable system either in the IE or IR domain.

2.5 Summary of Table Models

The above review of table related work underlines the importance of some form of table model. As with models of language, any declaration of a model of tables will be contested. However, it is clear from the above that there are a number of different levels at which table research appears to operate. These 'levels' rarely, or only to a limited extent, interact though some overarching description of a table must account for this prior art. A synthesis of the different levels at which tables are characterised for particular research tasks is presented in Figure 2.1, and it is this which forms the basic outline of the model evolved in the second part of this thesis.

[2]The U.S. Securities Exchange Commision (SEC) receives reports filed by all public companies in the United States of America.

- table
  - physical
    - * spatial
    - * procedural
    - * declarative
  - functional
    - * simple (head and body)
    - * complex (index and data)
  - structural
    - * one hierarchy
    - * multiple hierarchies
  - semantic
    - * category
      - · data driven
      - · semantics driven

Figure 2.1: Levels of table models

2.6 Chapter Summary

This chapter has overviewed the main areas of research connected with tables in general. Further discussions on related work can be found in Section 1.3 where IE systems are discussed, and Section 3.1 where IE systems involving tables are described. In addition, Section 4.5.7 describes some work on theories of context and ellipsis and their relevance to the work presented in this thesis.

For comparison, [LN99] is a useful publications surveying the field of table processing.

Chapter 3

Tables and Information Extraction

This chapter looks at table processing for information extraction. It also refines the class of document objects called tables.

3.1 Table Modelling for Information Extraction: Discussion

It is perhaps of some note that in many articles pertaining to tables, their use and computational and cognitive processes associated with them, the class of document objects termed 'tables' is often not defined, even in informal terms. One of the reasons for this is a lack of a definition of the underlying conceptual elements which go to make up tables — just as we require some model of the type and role of linguistic elements to distinguish a sentence from a random stream of words, so we require some language to describe the basic components of non-linear textual objects and their relationships in order to distinguish tables from, say, lists.

In the field of tables and IE (T/IE), there are perhaps only three reports of viable systems: Laurentini and Viada ([LV92]), Shamillian *et al* ([SBW97]) and Green and Krishnamoorthy ([GK95a])[1]. Laurentini *et al* describe the class of tables with a very limiting descriptive terminology, whereas Shamillian *et al* propose a very limited view

[1] In [HAB+], SRI International refer to a system capable of dealing with tables, however the publication which was 'to appear' has not yet (as of June 1999) been published.

of what may be a member of the class of document elements called tables (essentially restricting the interpretation of a table to a set of uniform records). Green *et al* provide a concise definition of the aims of their system which indicates separate physical and logical table components, though fail to recognise both the potential complexities of the relationship between the logical table and the physical table and the ambiguities that the physical table presents which need further, linguistic, processing to identify and resolve (see **Table (4.7)** and **Table (4.8)**).

Often, the tables appearing in the literature are of a very varied nature (the variation between tables in one piece of research is small, however between different researchers it may be quite broad; see, for example, the tables appearing in [GWK93]). This leads one to believe that certain restrictive definitions appear due to specific application definitions, and specific domains in which these applications are deployed; outweighing the need to provide a general description[2].

For example:

> By tables we mean a class of page layouts in which the data are presented in 'record' text lines made up of fixed-width fields containing characters. [SBW97]

which is based on physical templates — i.e. a set mappings from the relative location of page areas to a model of tables; and:

> From a logical point of view, we consider a table as a set of elements T_{ij}, arranged in i rows and j columns. The elements T_{ij} are usually text blocks, but in some cases they also consist of other objects like drawings, pictures or mathematical formulae. A closer look very often allows us to perceive a table as one relation of a relational data base. ([LV92])

This is an abstraction of the physical table, not a description of the table in any logical form. Green and Krishnamoorthy offer a similar description:

> A printed table is the visual manifestation of a logical relation. ([GK95a])

This basic description is enlarged upon by Green:

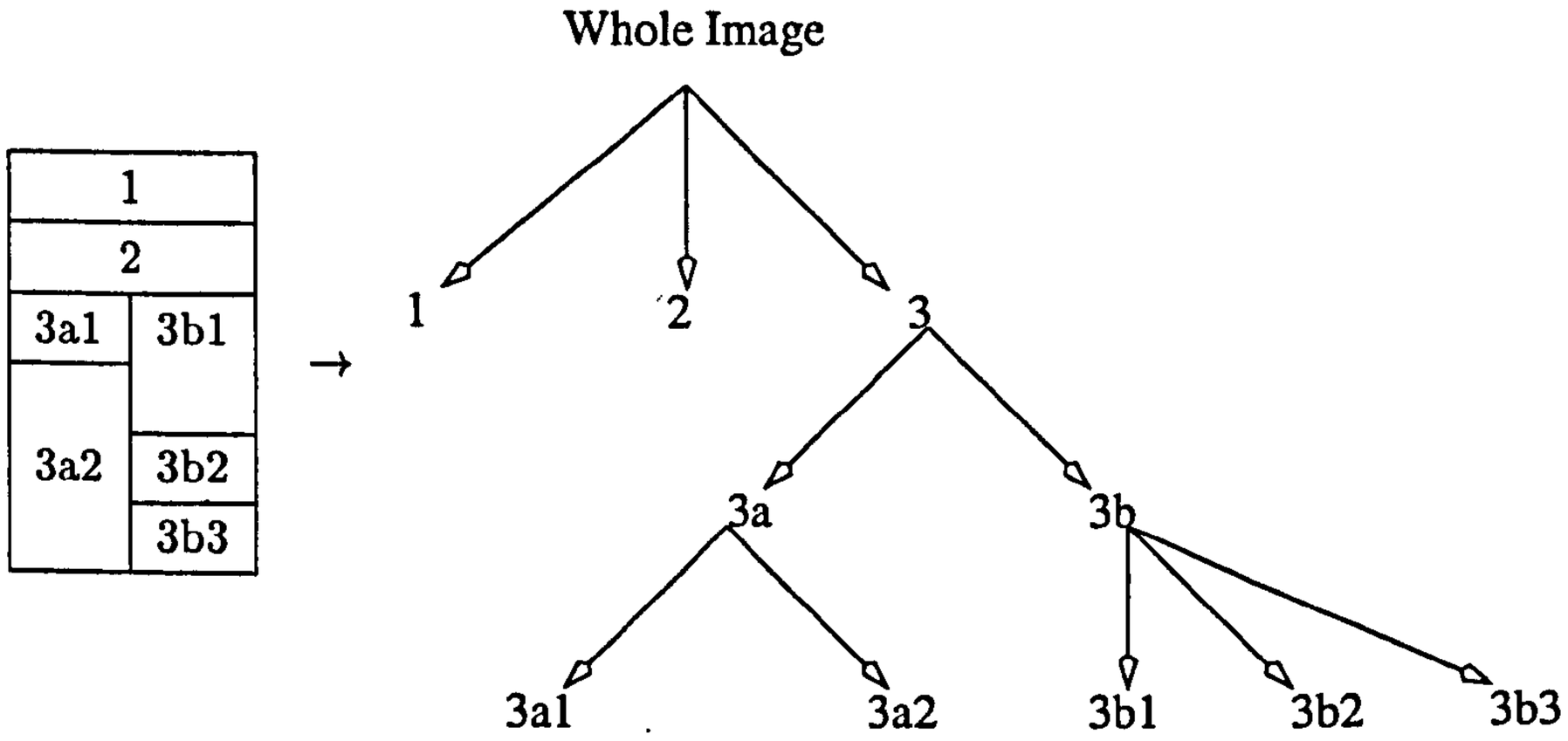
[2]This is not a criticism. If you require a system to deal with only one format of table then you are entitled to define that to be your class of objects called 'tables'.

> Tables are rectangular arrays of image space within which information in row and column regions are related in some way. It is convenient to think of two types of tables, physical tables and logical tables. Physical tables are the printed manifestation of relational information. Logical tables are "relations", in a relational database sense (in fact, relations are called tables in SQL). It is a common practice to combine more than one relation via merges and joins, in the preparation of generating the data prior to printing it; thus a printed table may represent more than one relation. Also, the same relation or set of relations will have many possible physical table layouts. ([Gre97])

The difficulty in providing, and the lack of, concise analytical descriptions of tables is perhaps best summarised by Wang (and which is echoed by Loprest and Nagy in [LN99]):

> It may be easy to point out a table in a book, but a precise definition of a table is elusive. ([Wan96], p. 2)

A presentation commonly used in the low-level physical description of tables and that used by Green ([Gre97]) which describes the table using XY trees[3]. The following example appears as an illustration of the XY tree concept.



[3]Green accepts that there are certain arrangements of tables which cannot be captured by this model

The table is considered in alternate horizontal and vertical (X and Y) divisions. Clearly, this representation cannot account for any ambiguity between the appearance of the table and the logical structure of the table. The XY tree is effectively an abstraction of the physical relationships (i.e. 'adjacency', 'spanning'[4] and so on) between cells, not the logical relationships. The differences between **Table (4.7)** and **Table (4.8)** can never be encoded by this formalism.

He states:

> The goal of this research [into the analysis of tables] is simply to drive out the relations given the physical table. ([Gre97])

which bears some similarity to the goals of this work. However, Green uses only the physical features of the table to 'drive out' the relations in the table. The underlying assumptions here mean that no physical arrangement of cells, when described by the XY tree, is ambiguous, and that there is no variation in the manner in which a 'relation' can be presented.

In addition, his investigation into the nature of the relations in the table fails to be in line with one of the key points of the relational database model: the dependency between data and the use of multiple linked tables to express this . In other words, if a comparison must be drawn between a table in a document and a table in a relational database, it must take account of the fact that the document table is capable of representing some number of views of a set of relational database tables *simultaneously.* In this sense, the view is a function which combines and filters information in a set of tables. The issue is further complicated by the fact that a view of a relational database must still be a valid relational database table; document tables exhibit quit different and more complicated indexing structures (the head and stub). In addition, a system which tries to derive the 'relations' in a table by simply establishing which 'rows' are to be converted into some sort of tuple misses the issues of the complexity of the stub (see Figure 3.2) which may be of arbitrary width (and is to some extent analogous with the key field in a database table but far more complicated). In other words, although some concession is made to the fact that a document table may represent some number of relational tables, or a combined view of those tables, no account is made for the complexity that this implies, in particular in the head and stub of the table which *do not* necessarily appear as the simple

[4]These terms are described in full in Chapter 4.

Year	Term	
1991	Winter	
	Spring	
	Fall	
1992	Winter	
	Spring	
	Fall	

(Year, {(1991, ∅), (1992, ∅)}),
(Term, {(Winter, ∅), (Spring, ∅), (Fall, ∅)})

Figure 3.1: Wang's Abstract Tables. The abstract category, presented as a tree-like structure, will be employed later in the development of the table model (Chapter 4).

indexing structures found in relational database tables. The scope of the complexity of the interaction between the relation of the database type and the presentation of many relations in the document table is discussed in Section 4.5.

Pyreddy and Bruce ([PC97]) offer an account of the table based on lines of text as in an ascii document. This model mixes syntax and semantics. Firstly, it performs text zoning to establish the 'type' of lines in a document (table or not). Lines might be paragraphs, headers or tables. Then, for the table lines, it produces a semantic description by assigning certain 'roles' to those table lines. Table lines might be data or header (what they term 'captions'). However, such table zoning cannot really be regarded as structural and is more akin to the functional description of tables discussed later in this thesis (Section 4.3).

Finally, the most useful description of the logical table, and the one which goes the furthest in separating the physical from the logical table is that of Wang ([Wan96], [WW93]). She provides a hierarchical description of multiple categories which is independent from the kind of structural hierarchy that is strongly tied to the physical table as presented by Green and Krishnamoorthy.

Figure 3.1 shows an example of a table fragment and the notation used by Wang to describe the categories within the table. Note that the repeated material in the column labeled `Term` is not repeated in the abstract table as represented on the right of the figure. Wang's representation also includes a mapping from the categories to the individual data cells contained in the table.

The separation of these abstract categories from the structural view of the table (having one or two hierarchical structures) is key to getting at the dependencies and relationships between the cells and their contents.

The purpose of this thesis is to provide a model of a table suitable for the task of information extraction. In order to do this it must define the *desiderata* for a model, and then describe a model which fulfills them. This task can be achieved by discussing the following:

1. Define what we mean by a table (Section 3.1.1, Section 3.1.2).
2. Define, in abstract terms, what a model of a table must look like (Section 3.1.3).
3. Define the task/application that this model is to operate in and consider the requirements of that process with respect to any possible model (Section 3.1.4).

Resulting from this discussion, the following task can be attacked:

> Advance a model of tables for information extraction by fully describing an instance of the general model (Part II).

3.1.1 What is a Table?

First, a graphic description of a table is given in terms of its appearance as an element in a document. Secondly, the table is considered as an information bearing component of a document. Finally, we look at the table as it functions in a document[5].

The Table as a Physical Document Element

A table is an area of a document which is characterised physically by a grid-like appearance. This grid is often expressed by some amount of line-art, though tables with little or no line-art are also often encountered, which raises issues for table recognition and structure recognition (e.g. [KD98], see also Section 2.1). The table is a static component, unlike a form, and as such is not to be altered by the reader. The areas indicated by the grid (either singly or in aggregate) contain material which may be textual, ideographic, graphical or any other printable or displayable form. The table may be in place or a floating element. We do not limit our definition of a table to a simple relational list (as does, for example, [SBW97]), though this is a common

[5] c.f. [Wan96]: pp. 2-7.

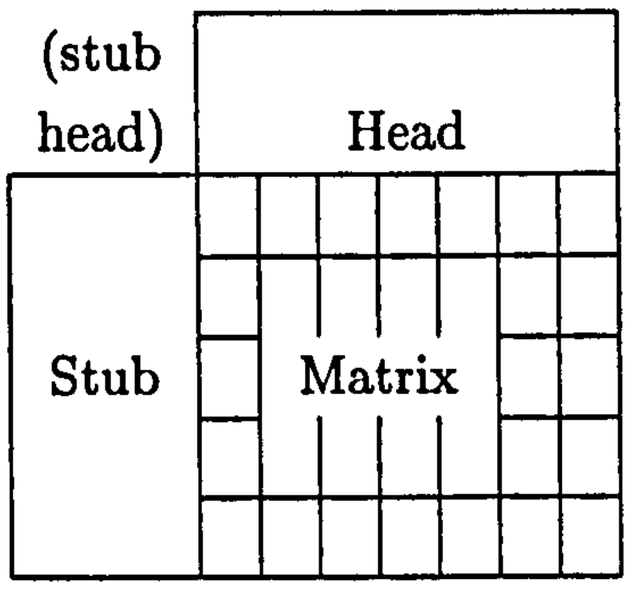


Figure 3.2: Common table terminology

form, but allow tables that might in general be described by the nomenclature of, for example, The Chicago Manual of Style ([sty93]), or Wang's thesis ([Wan96]). The basic terminology introduced in those publications is shown in Figure 3.2, and further complexities will be introduced in Chapter 4.

It is also possible to classify tables according to certain functional or physical characteristics (e.g. the 'analytic', 'reference', 'archival', 'raw data' etc. tables described in [Lef89]). Again, in view of lack of consensus in this area (and lack of exposition in [Lef89]), we will not, for now, subscribe to any particular catalogue. Additionally, there are a number of conventional 'tables' such as the periodic table as well as a number of conventional formats such as the table used for competitive tournaments.[6] We take no special account of those here and submit them to our general model.

The table as a whole can be broken down into a number of general components as in Figure 3.2 (following [sty93]). The cell is the smallest such component and is the basic currency of the physical table. The stub is the left hand portion of the table which is, specifically in a matrix table, used to index the content area. The head is the uppermost region which is, again, used to index the content area. Additional terminology will be introduced later where appropriate.

[6] Sometimes referred to as a round-robin.

Tables as Information-Bearing Document Elements in a Discourse Context

Tables are not objects which we interpret in isolation. Like any other information-bearing component of a document, they are to be understood in context. However, understanding how a table is *created*, or even *why* the author used a table rather than a graph or simple prose to convey the information is not a topic within the scope of this work (though a summary of the literature is given in Chapter 2, and some of the concepts presented in that field are used to construct the model presented in this thesis).

We take the view that the table is an arrangement of cells which have content. The cognitive mechanisms which decided what that content should be are not examined here, though of course, such information would be useful for our purposes.[7] Rather, we assume that there is some information that the author wishes to impart, this information is expressed by the content of the table's elements *and* the organisation[8] of those elements.

The table has some links with the text in which it appears. Firstly, the table may be referred to. Table reference takes two general forms.

- **explicit reference.** In this case, the text refers to the table by the use of some specific unique string (explicitly quoting the name of the table generally using the numeric or other indexing scheme employed by the document). For example `the results are shown in Table 2.2.`

- **implicit reference.** Here, the table is referred to in a less direct manner, without the use of a unique string. For example, `the results are shown in the following table, the results are shown below.` This form of reference may use a logical locator (like `below, above, at the end of the document` etc). In other cases, a content based description is given (`the table of results shows that`).

---

[7] We take the set of cells as a given, though in fact it seems clear that this is not the case: The act of creating the table will undoubtably influence the content and the organisation of the cells.

[8] We use the term 'organisation' to indicate the relative arrangement of cells in the table. This identification of meaningful and non-information bearing organisation will be discussed in Section 4.4.

gram	pre			post		
	#	string		#	string	
1	36			39		
	29	are shown in		10	1	
	7	is shown in		8	2 to 9	
	6	file class in		7	3	
	4	in the following		6	4	
	4	in the appropriate		6	2	
	4	dimensions given in		5	10	
	4	are presented in		4	6	
	3	results presented in		4	5	
	3	of the truth		3	of results	
	3	in		3	by an amount	
	3	as shown in		3	below	
	3	the following		3	9	
	3	1 & 1986		3	12	
2	36			39		
	8	class are shown in		10	1	
	5	results are shown in		7	3	
	4	the dimensions given in		7	2 to 9	
	3	part 1 & 1986		6	4	
	3	in		6	2	
	3	the following		5	10	
				4	6	
				4	5	
				3	of results	
				3	by an amount more	
				3	below	
				3	9	
				3	12	
3	114	in		39		
	36			28	2	
	16	the		25	1	
	13	truth		18	4	
	10	following		13	3	
	6	parsing		11	for	
	6	and		9	of	
	5	of		9	5	
	5	appropriate		7	6	
	5	1986		7	10	
	5	,		6	should	
	3	see		6	below	
	3	prevalence		5	i	
				4	shows	
				4	is	
				4	7	
				3	by	
				3	9	
				3	8	
				3	12	
				3	11	
4	45	shown in		39		
	36			10	1	
	10	the following		8	2 to	
	9	presented in		7	3	
	7	the truth		6	4	
	7	given in		6	2	
	6	listed in		5	10	
	6	class in		4	of results	
	5	in the		4	6	
	5	: 1986		4	5	
	4	the appropriate		3	should be	
	4	specified in		3	of contents	
	3	summarized in		3	by an	
	3	lr parsing		3	below	
	3	from the		3	9	
	3	a truth		3	4 and	
	3	(see		3	2 shows	
	3	in		3	2 presents	
				3	2 and	
				3	12	

Figure 3.3: Context for the word `table` in the table corpus. Empty strings in the pre or post positions refer to the beginning or end of a sentence.

It is possible to explore the corpus of tables and the documents which contain them to discover what phrases are used to introduce tables (see Figure 3.3).

Another key relationship between the table and the text is the discussion of the contents of the table and the summarisation of information or the conclusions which are drawn from the table as a whole.

Meta-text of one form or another is used to describe how to read a table. It may clarify the terms used (`DTD # refers to the number of the document type description`), or provide some indication of restriction (`the results are the maximum found`).

Finally, there are similar explanatory relationships between the title and the table, and the caption and the table, e.g. `Table 5, Results for Experiment 1 expressed as logarithmic values.`. These portions of text may also indicate how the table is read, what the relationships are between entries and so on.

The content of titles and captions varies from a description of the focus of the table to an extra categorisation of the information. This categorisation distinguishes the information in a particular table from that in others which have very similar categories and structure; e.g. those which form a set of results for different experiments or investigations.

The Table's Function and Use in a Document

We take the use of a table by a reader as having two basic forms:

1. Accessing an individual cell (*cf* 'local search' of [GWK93], p. 189).

2. Comparing a number of accessed cells (*cf* 'global search' of [GWK93], p. 190).

Local search is demonstrated by the reading of the cell containing `5,514,000` in the table below (**Table (3.1)**). Comparing this cell with that containing `7, 855, 000`, as might be done when comparing the data for different years, is an example of global search.

(3.1)

Parties	1923		1924	
	Votes	Seats	Votes	Seats
Conservative	5,514,000	257	7,855,000	419
Liberal	4,265,000	158	2,985,000	40
Labour	4,358,000	192	5,482,000	151

Local search (for tables) is the act of locating a single piece of information in a table. This amounts to locating a single cell. **Global search** is the act of deducing new information from a set of local search operations.

This categorisation sits well with the cognitive processes associated with table reading as described by Wang in [Wan96], p. 5: the comprehension of organisational principles and the underlying logical structure[9], a *search process for locating relevant information* and an *interpretive and comparative process.* In this thesis we will look at the first only: *local search.*

In addition to the above aspects of the table's use in isolation, we can consider the place the table holds in the discourse or rhetorical structure of the document. For the present, we note that there is some indication that analyses such as those summarised in [KD94] (and exploiting such data as presented in Figure 3.3) are appropriate for the manner in which tables are introduced and discussed in the context of the document as a whole.

Summary Overview

A useful summary is presented in [Cam89], chapter 2 describes 'Essential Characteristics of Tables' (p. 5)

> A *table* is an object which uses *linear visual cues* to *simultaneously* describe *logical connections* between the *discrete content entries* in the table. A *content entry* is the basic component of information in the table. Basically, a content entry can be any visual symbol. Note also that the *content entries* of a table are *discrete*, that is, each content entry of a table is clearly separated from every other *content entry.* ... A *logical connection* refers to any human derived mental connection. Logical con-

[9]The logical structure here, I think, is the organisation of 'categories' in the table: i.e. the groups of interdependent cells. That is, the number of dimensions as in a chart or graph or other graphical device. [Wan96], p. 3: The tabular items and their logical relationships provide the *logical structure* of the table and the number of categories defines the logical dimensions of the table. The term 'categories' is not further defined in [Wan96] with respect to the semantics of the cell *contents*, though some examples are given. Additionally, there appears to be no exposition of 'logical relationships' and consequently, the term 'logical structure' lacks a well defined meaning. It is supposed that these terms are in some manner accepted terminology as they appear in much table related literature, though again, without clear definition.

nections can be very concise, e.g. students in a class that I teach, or they can be very general, e.g. things I like to do.

A *linear visual cue* simply means that a logical connection is implied between table *content entries* that are viewed as arranged along a linear axis.

this summary continues on p. 8 with

> Other table models break down the *content entries* in tables into data *content entries* and heading or label *content entries.* The idea behind label entries is that they are used as indexes for information in the table ([Bea85], [Pub86]).

Perhaps key to this definition is the notion that the visual cues *simultaneously* describe *logical connections* between cells, though this is not the only manner in which connections may be established or indicated.

3.1.2 The Logical Document and a Refined View of Tables

The above section has introduced certain dimensions along which we may characterise tables (physical objects, discourse objects), in particular with respect to already existing models and descriptions. However, in order to advance the definition of a table in the context in which we intend to model it, it is appropriate to give an outline of the document in terms of logical organisation and then to further refine the above concepts and focus the notion of a table. In fact, what is required is that we provide a description of the features of information presented in a table-like manner and use these features to define the class of objects which we are interested in. This class of document elements we term 'tables'. However it may exclude certain examples included by other definitions.

A gross model of a document in terms of the organisation of the content can be given by employing two key concepts:

1. Hierarchy: inferior/superior relationships.

2. Order: a linear sequence of objects.

These concepts are used at many different levels of description including physical descriptions and logical descriptions. In the case of logical descriptions the *hierarchical* nature of a document is used to provide specialisation in the exposition of concepts topical to the document: a section heading indicates a general concept which is further refined by subsection headings and so on. We say that the section **dominates** the subsections below it. Because tables often contain broad hierarchical structures, i.e. labels which dominate many inferior cells, when discussing this hierarchical relationship, the term **distribution** is also used. The semantic mechanisms which provide relationships between these units and their interaction with discourse models are not discussed here, though it is straightforward to imagine some possible examples.[10] The *ordering* of the elements is often employed to sequence elements of an argument or description in the usual communicative manner.

As noted elsewhere (Chapter 6), the table, when a certain level of complexity is being modelled, cannot be described in the in-line manner in which we are accustomed to reading documents. This is due to the multi-dimensional hierarchical nature of the relationships being expressed (Section 3.1.1). Consequently we encounter not just one continuous hierarchy of distribution as we see in the document as a whole, but two or more. Note that distribution may be considered from a single node (cell distributed over its children) or from a number of equivalent nodes (siblings or orphan siblings[11]).

In the following example, there are two such hierarchies of distribution and one orthogonal distribution between these two hierarchies. `Products` are listed next to their `Values`.

(!3.2)

Product	Value
Guitar	$ 100
CD Player	$ 120
Telephone	$ 50

The `Product` hierarchy acts as an index in to the monetary `Values`. It is part of the indexing or access structure. The cell containing the string `Value` is also part of

[10]The description of the tree structure of the document uses the term hierarchical simply due to the logical tree like connotations of this term. A more abstract term might be a 'head and dependency tree'.

[11]Terminology describing the relationships between points in a hierarchy often take the form of familial relation names, e.g. parent, child, sibling, etc.

the access structure. It has a slightly special role as it is part of the access structure but also the root of the category representing the data of the table: its values, the hierarchical leaves, are the data of the table, the information which is found in a local search. A category which is wholy or partially in the access structure of the table is termed an **access category**. Cases also exist in which the data of the table are not set in a category that is part of the access structure. The two hierarchies are examples of what are called categories. The distribution from the set of `Products` to the set of `Values` is an example of **intersection**. Intersection is the indexing of one category by another.

It should be noted at this point that the elements found in the hierarchies which we term categories are strings, and not some interpretation of the string contents of cells. However, the relationships which we (here informally) discuss as being recognised between the elements in the hierarchy can be viewed as acting between some form of interpretation. In the majority of cases this apparent contradiction is transparent as the strings found are simple noun phrases and we can easily substitute some similar orthographic representation of the meaning into the category. However, there is no restriction to having noun phrases, or any other simple syntactic object in a cell. It should also be noted that the strings in the hierarchies are actually tokens which represent the strings in the table. Consequently, when a category is recapitulated (a term which will be introduced in full later, but which implies the repeating of category strings in a set of cells), equal (or equivalent) strings appear in the table but have only a single unique representation in the category.

The minimum arrangement for a table is that in which two access categories provide at least one dimension of distribution each, and the leaves of at least one category index into the data in the table. Distribution is further illustrated in the following example (**Table (3.3)**).

(!3.3)

Label	**Label**
Sub-label	DATA
Sub-label	DATA

In **Table (3.3)**, the `Labels` distribute over either `Sub-labels` or DATA. The `Sub-labels` in turn distribute over DATA (as the `Sub-labels` are siblings — their relationship to the parent is the same — we consider the distribution to be from this *set* of cells over the DATA). In this table, bold font indicates the indexing or access area or structure of the table.

However, distribution is not simply a physical feature of a table, but a logical hierarchical feature. In more complex situations, such as the following (**Table (3.4)**), the notion of distribution can be seen to be related to more abstract qualities of the information presented in the tables; and not a simple mapping from physical features to logical description.

(3.4)

Animal Type
Dairy
Beef
Veal
Swine
Growing pig
Mature hog
Sow & litter
Sheep
Goat
Poultry
Layers

Here, `Animal Type` distributes over `Dairy`, `Beef`, `Veal`, `Swine` and `Poultry`. It doesn't distribute over `Growing Pig`, `Mature hog`, `Sow`, `Sheep`, `Goat` or `Layers`. Rather, these elements are distributed over by the cells containing the strings `Swine` and `Poultry`. The entire hierarchy is an example of a category.

A **category** is a hierarchy of cell content strings. Any cell content strings immediately linked in the hierarchy stand in some semantic relationship with each other. All the semantic relationships in a single category are equal. For example, in **Table (3.4)** all the cell content strings represent classes of farm animals and the relationship represented by the parent-child link is that which might be termed 'type of'.

The distribution described above between a cell and a set of cells or between two sets of cells occurs either within a category, or across the leaf cells in a category.

It is this notion of distribution which we use to refine our notion of what is a table and what is simply formatted in a tabular manner. A table, as we see it, must contain at least two categories and at least two examples of distribution. One of these examples of distribution must involve category leaves (intersection). This definition rejects lists and enumerations (**Example (3.8)**, **Example (3.6)**). It also rules

out assignments with generalised value cells under-spanning attribute descriptions (**Example (3.19)**).

A category does not need to have a label or any explicit structure. Some unlabled categories are *implied* by the presentation of the table. For example, the abstract table **Table (3.3)** has a category containing the text DATA. If there where no clear relationship[12] between the contents of these cells and the `Labels` above or the `Sub-labels` to the left then this would be an implied category. **An implied category** is one which consists of only a set of leaf nodes and has no internal structure. In the example below (**Table (3.5)**), the matrix category is an implied category. The values for expenditure are not related in any clear hierarchical manner to the elements in the stub nor to the elements in the head, but are rather an individual domain of information (the **intersection domain**) which is intersected by the two other domains: {Matt, Wakako, Pete} and {Food, Entertainment}. Intersection can be viewed as being similar to implication: $a \in D_0 \rightarrow x \in D_1$; or from the following example: `Matt` $\in D_{people} \wedge$ `Food` $\in D_{expense\_categories} \rightarrow$ \$10.00 $\in D_{expense\_value}$.

(!3.5)

	Food	Entertainment
Matt	\$ 10.00	\$ 5.00
Wakako	\$ 7.00	\$ 5.00
Pete	\$ 8.00	\$ 15.00

A path through a category is a path from the root node of the category's hierarchy to one of the leaves. For example, in the animal example (**Table (3.4)**), `Animal Type`↷`Swine`↷`Growing Pig` is a path through the category. Reading a table is

[12] A clear relationship is a key part of the concept of the category, yet is a very difficult concept to define. In the simple cases, the term is easily applied (for example, the type of hierarchy in the animal type example). However, as this term is used to distinguish a category from cases in a table where we would rather separate a structure whose physical cues might indicate a category, but which make more sense as at least two categories (for example an implied category and an access category) then the term is possibly contentious and at best seen as being some relative description of how 'intuitive' the relationship between elements in a possible category might appear. It is often the case that the strings in the data area of the table exist in an implied category and do not have a 'clear relationship' with the indexing structure above or to the left of it, however, as shown in this discussion, this is not always the case. As Cameron suggested ([Cam89],), the *logical connections* can be very concise or very general. The notion under discussion here is one which suggests that the more concise, intuitive, world knowledge or natural logical connection between strings forms the categories, and the other categories exist at the point of intersection between categories.

essentially a matter of combining paths through categories. The structure of the table indicates when these category paths intersect. This is generally found in a category at the centre of the table: the matrix category.

Cases clearly arise when the intersection of categories is not appropriate due to missing data or other factors. In extreme cases, the categories involved in the intersection can be partitioned and mapped to mutually exclusive subsets of the intersection category. In these circumstances, the table can often be split into sub-tables. These sub-tables must then be re-examined. The following series of examples illustrates the notion of category intersection and its relevance to the classification of table-like document elements.

Example **Example (3.6)** represents a simple list.

(!3.6)

$item_0$
$item_1$
$item_2$
$item_3$

In **Example (3.6)** there are no intersections, nor is there any distribution. The list items form a category, but there is no hierarchical structure. The next step up in complexity is represented by the labeled list in example **Example (3.7)**.

(!3.7)

List
$item_0$
$item_1$
$item_2$
$item_3$

In **Example (3.7)** there is a single distribution (the label over the list elements) but there is no intersection as there is only one category. We can continue with more complex forms of lists. For example, **Example (3.8)** appears to be a table, however it is in fact a list as there are no intersections because there is only one category.

(!3.8)

Cars	
Ford	Honda
Ford1	Honda1
Ford2	Honda2

However, it may be argued that we can present the same information as a table, in which case this definition of a table is purely to do with the manner in which information is presented. However, examples such as the following (**Example (3.9)**) still demonstrate a lack of intersecting categories.

(!3.9)

	Cars	
	Ford	Honda
Ford1	X	
Ford2	X	
Honda1		X
Honda1		X

This is a good example of the type of objects lying on the boundary that we are attempting to draw between some theoretical notion of a table, which may be characterised by a complete model, and the more general idea of the arrangement of text or other document elements on the page in a tabular manner. Clearly, there is some form of organization present in **Example (3.9)**, though it employs a redundant 'category' (the boolean category represented by **X** or the absence of **X**). However, it is claimed that the use of the boolean category is more diagramatical than textual (this is not to say that such a mechanism is useless, clearly the organization *is* useful). It *indicates* the relationship between two 'things' but does not add content to the description. A simple flat list of items may be augmented be stating that each element in the list is in the list using a similar strategy (**Example (3.10)**).

(!3.10)

$item_0$	X
$item_1$	X
$item_2$	X
$item_3$	X

Should this, and its potential extensions, be included in the class we are wishing to define?

Example (3.9) can be partitioned in to two sub-tables (**Example (3.11)** and **Example (3.12)**):

(!3.11)

	Cars
	Ford
Ford1	X
Ford2	X

(!3.12)

	Cars
	Honda
Honda1	X
Honda2	X

Such table-like objects are poorly designed mechanisms for displaying the information and can be simply collapsed to the list form (**Example (3.13)**).

(!3.13)

Cars
Honda
Honda1
Honda2

In addition, if we take the final rearrangement of **Example (3.9)**, we can see that the use of a spanning cell which is not dominating a portion of the indexing structure — i.e. a cell in the head which spans the matrix category but doesn't span any further material in the head — is actually a title and should not be considered as part of the structure of the table. In fact, if it were considered as a root of the matrix category, then it can be seen that it distributes over the cells in two dimensions. This is not the normal behaviour of a cell in a table, and is indicative of the table title or caption.

(!3.14)

	Cars	
Ford	Ford 1	Ford2
Honda	Honda1	Honda2

The above discussion and examples have been useful in aiding us in locating the area in which most effort is needed in providing precisely what a table is, as distinct from tabular arrangements of text. Certain cases may fall on either side of any definition and so continued discussion in the field is perhaps best focused here.

A further example can be used to introduce the notion of recapitulation (for further discussion refer to Appendix A). Compare **Table (3.15)** below with **Table (3.16)**.

(!3.15)

A			
X		Y	
1	2	3	4

(!3.16)

A			
X		Y	
1	2	1	2

In **Table (3.16)**, there are two categories ({A.X, A.Y} and {1, 2}), whereas **Table (3.15)** has only one: {A.X.1, A.X.2, A.Y.3, A.Y.4}. The second, then, is a table according to our definition — it has at least two categories and an intersection category, whereas the first is a list. It will be seen in later examples that this notion of intersection of categories can influence the appearance and spatial efficiency of a table.

We also propose that a general semantic procedure for tables be advanced. This is the simple specialisation found in the reading paths: for a given cell the paths which intersect at its location are used to specify the type of the contents of the cell and provide an interpretation for them. For example, in table **Table (3.5)**, the interpretation of the cell containing the text `$ 10.00` is found by interpreting and combining the text for cells used to access it, i.e. `Matt` and `Food`.

Some uses of tables have become formalised with separate and particular semantics such as certain instances of truth tables, or unique table templates like the *periodic table*, and so on. A table obeying the general semantic rule lets us know where to find elements linked by a relation: on the reading path.

The following are examples of organised text which is considered not to fit the above criteria.

(3.17)

1 mil	0.001 inch	0.0254 millimeter
1 inch	1.000 mils	2.54 centimeters
12 inches	1 foot	0.3048 meter
3 feet	1 yard	0.9144 meters
5.5 yards or 16.5 feet	1 rod (or pole or perch)	1.6094 kilometers
1 mile	5,280 feet	1.6094 meters
40 rods	1 furlong	201.168 meters
8 furlongs	1 mile	1.6094 meters
3 miles	1 league	4.83 kilometers
	1 millimeter	0.03937 inch
10 millimeters	1 centimeter	0.3937 inch
10 centimeters	1 decimeter	3.3937 inches
10 decimeters	1 meter	39.37 inches or 3.2808 feet
10 meters	1 decameter	393.7 inches or 32.8083 feet

Though the text in **Example (3.17)** is tabular, the semantics of the table (the model by which relationships between cells is governed) is not the simple distributional model.

The following, **Example (3.18)**, is simply a list, i.e. one category.

(3.18)

Attribute
FileName
Owner
User
Developer
Size
Permission
AbsolutePass
Type
TimeStamp
LastAccessTime
i-nodeTimeStamp
Archive
Relationship
Use

Attributes and values presentation may use certain table-like physical devices like that in **Example (3.19)**.

(3.19)

7	6	5	4	3	2	1	0
unused			error				E

Example (3.19) is an attribute value list modified for presentational purposes.

Variety can also be found in the way in which table-like material is to be read. **Example (3.20)** is close to a paragraph in terms of the manner in which it is read.

(3.20)

The Hundred Years War	
The name conventionally applied to a period of intermittent Anglo-French struggle in pursuit of English claims to the French crown. After performing homage for his lands in Aquitaine to the King of France, the English King, Edward III, quarrelled with his overlord which led to open hostilities and in	
1339	Edward III proclaimed himself King of France, in right of his mother. There follow
1340	English victories at Sluys (naval, 1340) and Crécy (1346), and the capture of Calaise (1347).
1355-6	Raids by the Black Prince across France from south-west and French defeat at Poitiers.

Key to our classification of tables is the notion of relationships between cell elements. Consequently, the focus of the task of information extraction will be concerned with these relationships. A requirement for that is a model of the table which is capable of drawing out evidence of those relationships. While it is true that many (possibly the majority of) tables have a simple relationship between their appearance and the distributional semantics between cell elements, there are many cases where ambiguity arises. To give an initial example of the problem the first table below demonstrates the use of a spanning cell to indicate hierarchical superiority (distribution over) to the cells spanned.

(3.21)

Rank	Total in dollars		
	first weekend	first month	gross
1	Star Wars		
	100	200	500
2	ET: The Extraterrestrial		
	100	200	500

Here the 'category' `Total in dollars` is split into `first weekend`, `first month` and `gross`. The 'data' for these 'sub-categories' is placed in the cells which are perfectly aligned below (100, 200, 500, and so on).

Compared with the following, which still has a distributional semantics, but uses a physical template to connect the physical with the distributional model components.

(3.22)

Rank	Movie Title		
	first weekend	first month	gross
1	Star Wars		
	100	200	500
2	ET: The Extraterrestrial		
	100	200	500

To summarise the above situation, in the first table we can note that the numbers are likely to be dollars and so are related to the cell containing the string 'total in dollars' where as in the second 'Star Wars' and 'ET' have a relationship with 'Movie Title' whereas the integer cells don't.

3.1.3 Table Representation and Modelling: An Abstraction

We take a broad definition of a model to consist of two essential components.

1. An ontology.

2. A representation

The Model Ontology

We take the **ontology** of an object to be the initial analysis which subdivides the object conceptually. This provides a number of general, related aspects of the object. These aspects are distinguished in that they provide evidence for (or in analytical terms, constrict the search space of) each other. Perhaps the most immediate example is that of conventional linguistics which generally consists of an orthographic component, a morphological component, a syntactic component, a semantic component and possibly discourse and pragmatic components. Closer to the consideration of a table model, in OCR applications, the table consists of connected line segments

- which allows for patterns to be expressed in terms of combinations of lines - and textual or other areas (e.g. [GK95a]); markup models see the table as structured around groups of equivalent presentational format with some physical alignment (e.g. [Tho93a]).

In terms of applications, there will be a particular component of the ontology view which is relevant to the desired output of the application. Naturally, there must be an entry level component of the ontology which acts as the input to any computational system. These ontological components are not successive data types found in a pipeline of processes but are mutually supporting components. Ontological descriptions capture the aspects of the meaning of the table and the interaction of that meaning with the constraints imposed by other ontological components.

A key element in the design of a model with respect to a particular process is an appreciation of the progression of algorithms being applied to the model. We want to make sure that the model doesn't introduce unnecessary ambiguity or decision points. This is an effect of the model's representation and the meaning of the representation - or how we identify instances. For example, why don't we mark all tables with the same functional description which is in some way common to all tables (e.g. identifying only the header as distinct from the other material)? Because we would lose out later on the semantics: a functional description of the table that isolates the data from the indexing or access material removes a lot of effort from any model of the interaction between cell contents which would be implied by the simplistic functional model hypothesised above. We would also like to understand how errors in the model might propagate. Does an error at this component of the model introduce many errors later? For example, an error in the model of the table categories implies many errors for the functional model of the table whereas the reverse is not necessarily true.

The Model Representation

The representation is the component of the model which describes these conceptual subparts in a precise and unambiguous manner, providing a logical description. In addition, a model representation should describe how the model components are related in terms of the *constraints* that they impose on each other, thereby defining the space of possible tables.

3.1.4 Table Models and Information Extraction

To better understand how a model should be constructed, we must first characterise the application which is to exploit it. Having done that, we look at the implementation strategy appropriate to this characterisation and conclude with a description of the application's influence on the model design.

Information Extraction: Characterisation

The Information Extraction task requires that a predefined semantic template be constructed from free text. Generally, the task is restricted to a specific real world domain. However, in our specification of the task we consider the more general, unrestricted case. Free text implies that we cannot rely on rigid, symbolic processing and must introduce some degree of flexibility into the process. Unrestricted domains means that we must not rely on, for example, sublanguage lexical items and so on, or that we define clear ways to access them based on domain independent techniques.

However, this work approaches this general goal via a significant intermediate stage: the extraction of a table reading.[13] This table reading (the set of legal paths in a table) is augmented by a description of possible semantic relationships between those cell elements. The final stage, of asserting facts in some world model (instantiating the predefined semantic template), is a further goal beyond the reach of this work as described in this document. It requires the synthesis of the document interpretation with that of the table.

Information Extraction: Implementation Strategy

The degrees of freedom required by the task characterisation suggests an architecture with a little more flexibility than the standard IE implementation strategy of a cascaded set of transducers ([Hob93]). To allow for this flexibility, it is suggested that a quasi-blackboard design be adopted and that modules (transducers) post hypotheses on the blackboard which may in turn be accessed, assessed and modified by other modules. This reflects the interrelated nature of the ontological description.

The basic problem which we are to consider is the instantiation of a representation given a set of static knowledge resources, the results of previous analysis (i.e. the

[13] Table readings are defined in Chapter 4 and are essentially the cells encountered in a local search operation.

output of other modules) and a set of algorithms. The strategy is to establish under what circumstances a particular instantiation may be made. To do this we must look for patterns in the prior results which suggest the hypothesis, and account for ambiguities by exploiting the resources and prior results.

Application Influence on the Table Model

The robustness requirement indicates that we should provide a model of the table suitable for bottom up processing. This suggests that we start with an account of the physical table (the existence of cells and their relative position). As we desire to produce a description of the relationships between the cells in order to provide an account of the possible meaning of the table, the model must contain components of structure: i.e. where there are relationships, and semantics: what these relationships are.

3.1.5 Summary: Desiderata

A model of a table must contain

1. An ontology.

2. A representation.

In total, the model must clearly explain the features of each conceptual component, how these features might be recognised or extracted from a table and how the components are related. Here, a description of how the components are related is a description of what patterns and attributes found in one component of the ontology indicate, to some degree of confidence, the presence of an element in another ontological component. The representation must provide the necessary mathematical language to encode instances of the components. In addition, it must clearly define the model space of the tables via the constraints between model elements.

This can be fleshed out with respect to Wang's table model *desiderata*.

1. The model should capture a wide range of tables.

2. The model should be independent of the presentational form of the table.

3. A well defined mathematical representation should be used.

3.2 Summary of Category Models

Key to any true structural or organisational component of a table model is the notion of category. This term, is meaning and implications have been discussed at length above. Here we discuss its various forms in the literature.

Guthrie suggests a general definition of the category for document elements of any type ([Gut97]). Wood has a definition of categories which is more pragmatic. Wood suggests that the following table (**Table (3.23)**) has two categories X.[1, 2, 3] and Y.[a, b, c].

(!3.23)

X	Y
1	a
2	b
3	c

However, both Guthrie and Wood only offer intuitive accounts of what a category might be ([Woo], [Gut97]). Perhaps the most thorough account of categories is that found in [Wan96]. Though this is rather a model of the category, together with a set of operations which can be performed on the model for the purposes of table editing, it forms the inspiration for a component of the table presented later in this thesis (Section 4.5.2) and also provides a description of the table in general terms.

> The content of a table is a collection of interrelated items, which may be numbers, text, symbols, figures, mathematical equations, or even other tables. Some of the items are the basic data a table displays, and the others are the auxiliary data that are used to locate the basic data. We use the term *entries* to denote the former kind of data and the term *labels* to denote the latter kind. Labels are further classified into *categories* that are organised hierarchically. [Wan96], page 2

With respect to the model presented in this thesis, there are a number of important points to be made regarding the above.

- When discussing categories, Wang uses a natural language label to name them. This label may or may not appear in the table as part of the category hierarchy. Consequently, a model of the relational structure of tables which requires there

to be a label and values for the elements of the relation will fail in some tables where implied categories are found.

- Some categories are disjunctive in nature, a single path is selected and then used in conjunction with paths from other categories to index a cell in the data category. Other categories, as shown in Section 4.5.3, use paths *in conjunction*. These cases cannot be modelled by Wang's approach which essentially takes only one path from each category and maps that set of paths to a data cell.

- There are no descriptions of categories which indicate any conditions on the relationships between the nodes. We propose a minimal condition that all the relationships be the same throughout the category structure.

One of the aims of this thesis is to fill out a definition of the category (what will be called the data category due to the data driven methodology which motivates it) and algorithms for arriving at an instantiation.

3.3 Diagrams, Denotation and Tables

It is interesting to consider the relationship between tables and other complex document elements. The issue of the list and the table is discussed in Section 3.1.2; here the diagram is considered: we would like to consider if a table can be admitted to the class of objects termed diagrams. In [Ham95], Hammer provides an insightful commentary on the nature of diagrams and possible means for identifying and classifying them (although he states that there may be no clear set of features and criteria to distinguish diagrams from language).

One suggestion that he makes, which he later regards as inadequate[14] though it will serve here in summary, is that diagrams have

> [S]emantically relevant two-dimensional syntactic properties.

This comment is linked with the notion that the spatial arrangements of the components of a diagram are '*intimately connected to the relation expressed*'.

[14]It is deemed inadequate largely due to the fact that there is no requirement on diagrams to be two-dimensional.

As we consider a table to use spatial arrangement to indicate relationships, we might also consider them to be diagrams of something. Hammer also states the following.

> Diagrams can build the logic of what they represent into the physical logic of their grammar.

which is a suitable level on which to consider the case for tables.

It is certainly the case that the physical aspect of tables indicates certain types of relationships which might be expressed diagrammatically. For example, type of hierarchies, partitive hierarchies, instance of hierarchies etc. However, the number and variety of the possible relationships which exist between the elements of the table (in this case the contents of the cells) is large. In addition, the type of the relationships is not denoted by the indication of its existence, but is generally implied by certain aspects of the table *and document's* content.

To make an analogy with maps, it is as if all infrastructure were represented by lines between nodes (e.g. cities, towns, villages). The nodes are all labeled in the same font regardless of their type and the type of the infrastructure can only be deduced if the type of the nodes is known. In summary, the structural aspect of the table indicates the existence of *some* relationship, not the relationship itself.

An additional departure from the common conception of the diagram is the ambiguity imposed on the table's physical appearance by the number of dimensions of information in the table and the interaction between those categories. Consequently, the structure of the table which is only partially diagramatic, cannot be physically represented isomorphically in an unambiguous manner. Even the similarity between tables and Venn or Euler diagrams (*c.f.* the discussion on intersection in Section 3.1.2) cannot be carried through due to this ambiguity.

To classify tables as diagrams would require that we discard the idea that the spatial arrangements of diagrams are '*intimately connected to the relation expressed*'.

3.4 Chapter Summary

This chapter has summarised work in the T/IE area and discussed the range of document elements which appear and may be classified as tables. The concepts of distribution and intersection were introduced to clarify the essential nature of the class of document elements which this thesis will focus on.

Summary of Part I

Part I motivated the goals of the thesis and summarized the relevant research fields. It demonstrated that although there is a broad spectrum of work being carried out on tables (Chapter 2) none of this work has established a general model of tables. The models of tables presented in publications from the various fields form the motivation for the model presented later in the thesis. A summary history of information extraction (IE) was presented (Chapter 1, Section 1.3) and the utility of exploiting documents containing tabular information was discussed.

The goals of the thesis were refined through a discussion of the potential interaction between table processing and a typical information extraction system (Chapter 1, Section 1.4). Further investigation of the prior art in tables for information extraction (Chapter 3, Section 3.1) demonstrated the need for a definition of the class of document elements called tables (Chapter 3, Section 3.1.1) and a presentation of table modeling (Chapter 3, Section 3.1.3) completed the background and motivational sections of the thesis.

Part II

A Model of Tables

Q4

Part II introduces and defines the model of tables. Chapter 4 provides a discursive account of the phenomena found in tables and the relationships between components of the table model which can be used to identify and describe the ambiguities found in tables. In particular, the relationship between what appears on the page and the underlying logical structure and meaning of the table is presented with a catalogue of those physical table elements.

Chapter 5 provides a formal description of the model and introduces a symbolic representation of the table which can be used to describe algorithms in table processing systems.

Chapter 4

The Model Ontology

This chapter introduces the components of the table model. Each component is motivated and discussed in detail. The discussion indicates why the component is required and what issues of ambiguity arise from it with respect to the other model components.

4.1 A Model of Tables for Information Extraction: Overview

The model proposed here has the following components:

1. Graphical: This thesis assumes some basic graphical representation of the table, e.g. a bitmap of a document image.

2. Physical: a description of the table in terms of the physical relationships between its basic elements when rendered on the page.

3. Functional: the purpose of areas of the table with respect to the use of the table by the reader.

4. Structural: the organisation of cells as an indication of the relationships between them, the intent of the author and the restriction of the two dimensional page.

5. Semantic: the meaning of meta text in the cell, object text in the cell, the relationship between the interpretations of cell contents, the meaning of structure in the table and the meaning of a reading of the table.

Feature groups represented by degree 3	
(Exhaustive)	(Covisible)
176	122
63	46

Figure 4.1: *Cells are delimited, to various degrees, by line art, spacing and the interpretation of the contents.* The above table contains seven cells.

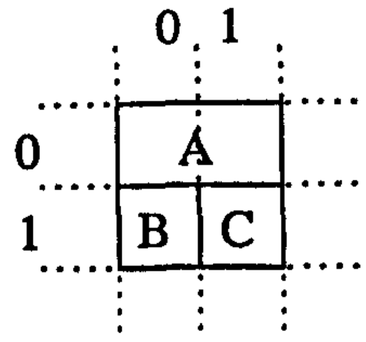
This follows the view of tables taken in [DHQ95] and builds on and extends some of the concepts described there.

4.2 Ontological Description: Physical

We take a reasonably high level view of the physical table (following [DHQ95] and [Cam89]) as containing a number of cells encoded in terms of *relative position* in the table. For our purposes, a **cell** takes the conventional meaning: an area within a table delimited either by the standard line art of table drawing or by assumed divisions in the table indicated by co-linearity of text, spacing and so on (Figure 4.1).

4.2.1 A Declarative Model of the Physical Table

The description of the physical table uses the relative position of cells as its basic informational element. In order to calculate the relative position of cells, we first superimpose a minimal grid over the table, and label the rows and columns with natural numbers. The cell is described by locating its top-left and bottom right corners in this grid structure. For example, the following table has a cell containing the string 'A' in the cell [(0, 0), (1, 0)], 'B' in [(0, 1), (0, 1)] and 'C' in [(1, 1), (1, 1)].



(!4.1)

This is the system proposed in [DHQ95] and prior to that, implied by the markup strategy presented in [Cam89]. Inspection of the physical model will tell us the relative position of cells — i.e. left of, above, to the right of etc. The **extent** (of a particular dimension) of a cell is the span of that cell in a particular dimension. When the horizontal or vertical extent of a cell either includes that of another or is equal to that of another the cells are said to be **aligned** (horizontally or vertically). In Figure 4.1, '63' and '46' are aligned horizontally, 'Feature groups represented by degree 3' and '63' are aligned vertically.

The physical description is enriched with some logical information: table footers, table headers, table captions and table labels.

This declarative approach to representing the physical layout of the table is intuitive in that its use of a coordinate system is all that is required to indicate the location and relative area of any grid of rectilinear shapes. However, it does limit the possible shape of cells to rectilinear ones, and consequently allows no explicit representation of alternative tilings involving triangles or parallelograms.

(!4.2)

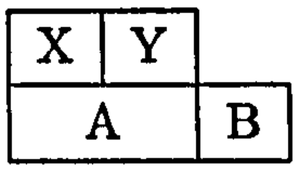
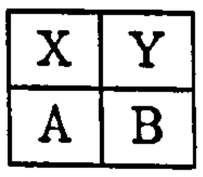
Column 1	Column 2	Column 3
1	2	3
4	5	6
7	8	9

We suggest, however, that the abstract qualities of these shapes can still be preserved by this encoding if handled properly. An additional abstraction assumed by the physical encoding is the orientation of the text in the cells. Since the physical component of the model simply encodes what the content is and nothing more, it doesn't say anything about the possible orientation of the text. It is possible to typeset text vertically and horizontally, as well as with any number of line breaks. Again, though information important to the meaning of the table may be encoded, or be apparent in these variations, they are not dealt with here.

The main advantage of this system is that it avoids any of the procedural baggage found in other systems such as HTML or LaTeX . In these table encoding systems, the nature of a particular cell can only be found by recording and understanding the *processes* which have been carried out for **rendering** (i.e. turning into a graphical

presentation) the cells which come before in the table — generally those cells above or to the left.

A simple example of this common to both HTML and LaTeX can be seen when a cell spans more than one column.

		
HTML	<TR> <TD> X </TD> <TD> Y </TD> </TR> <TR> <TD COLSPAN=2> A </TD> <TD> B </TD> </TR>	<TR> <TD> X </TD> <TD> Y </TD> </TR> <TR> <TD> A </TD> <TD> B </TD> </TR>
LaTeX	X & Y \\ \multicolumn{2}{|c|}{A} & B\\	X & Y \\ A & B\\

HTML describes rows in the table (TR). Each row contains a number of cells (table data — TD). A cell can indicate how many rows or columns it spans. This information is then used to calculate both the extent of the cell and the start position of any following cell. LaTeX uses a similar method of table encoding. Rows are described by a number of & separated cells. Cells which span multiple columns are indicated by the `multicolumn` macros. In both cases, from the examples above, the physical relationship to the cell containing B and any cell above or below it could never be computed without supplying an interpretation for the surrounding material (possibly the entire table).

These remarks are not criticisms of the procedural approach to table encoding as both the above mentioned systems are required to represent a table *for a certain task.*

The physical component of the table model can be considered, in some sense, to be the result of rendering all the other components of the table on to the page (in fact, the 'graphical model' is the final stage of the table generation process but this

is not discussed in this work). In particular, the relationship between the structural model and the physical model is discussed in Section A.2 which looks at the manner in which the structural table is realised, the physical vocabulary which is available and the information bearing and non-information bearing aspects of that process.

4.2.2 Justification, Font, Line-Art, Colour and Table Content

In transforming a model instance into a graphical presentation of the table — or taking a graphical presentation and deducing a model instance — there are a number of potential causes of ambiguity. We should consider the effects of the following typographic phenomena on the meaning of the table:

1. **Justification**: the alignment of the cell contents with respect to the cell space and other cell contents (e.g. decimal point alignment), and also alignment within the delineations implied by the superimposed grid.

2. **Font (and face)**: the use of font to indicate function (e.g. 'label' of a category).

3. **Line-Art**: the use of horizontal and vertical lines in the presentation of the table to indicate categories of information.

4. **Colour**: the use of colour, both for the presentation of text and as background or line colour, to indicate meaning and highlight information presented in the table.

The justification of cell contents is generally not significant as long as there is consistency. Even in the face of inconsistently justified categories it is unlikely to have any influence on the interpretation of the table. Justification is important, however, in deducing cell delimitation in the absence of any line-art, as the following SEC domain table indicates.[1]

[1]The SEC is a organ of the government of the United States of America which collects documentation produced by public companies concerning their buisness practices, economic history and plans.

(4.3)

	1993	1992	1991
Fuel expense	$102,670,217	$101,465,555	$ 93,686,895
Interest recoverable on deferred fuel and deferred purchased power costs -			
Recovered currently	(182,965)	1,328,931	2,439,668
Deferred for future return	461,058	(523,657)	(131,386)
Purchased power costs through the fuel cost adjustment	-	450,476	3,111,351
Reclass of sales for resale from purchased power capacity	-	72,399	1,139,399
Other fuel related items	15,378	(14,242)	25,315
Fuel revenue, as reported	$102,963,688	$102,779,462	$100,271,242

In **Table (4.3)** indentation within a single cell spaced column indicates some form of structure: `Interest recoverable on deferred.fuel and deferred purchased power costs-`, `Recovered currently`, etc.

Font and face distinctions, if present and used consistently may well indicate functional and structural information. For example, the use of bold face in the header and stub to distinguish these functional areas from the data in the centre of the table. In fact, font and face difference is very useful precisely when the expectations will give an incorrect interpretation of the table as in the following.

(4.4)

	SunOS 4	Solaris 2
	lpq [-P *printer*]	To examine the printer queue	lpstat -o
	lprm [-P *printer*] *jobnumber*	To cancel a print job	cancel *jobnumber*
	N/A	To move a print job to another printer	lp -i *jobnumber* -d *new-printer*

In **Table (4.4)**, the primary index to the information, the material that would normally be expected in the stub of the table, appears in the centre. It has been presented in bold face which attracts the reader to its location.

Thomas ([Tho93a], p 2.) makes some suggestions about the semantics of line-art, however the problem discussed is rather vaguely presented and there is no clarity regarding the functional or structural interpretation of the lines.

> ..., putting vertical lines between some columns and not others introduces a grouping which has semantic relevance. Changing such lines changes the way the data is interpreted. Thus the position of these lines contains semantic information and must be captured in any markup scheme which is intended to preserve meaning. [Tho93a], p 2

The positioning of the lines is still a stylistic factor and it is the meaning which those lines denote that must be encoded.

[Tho93a] does, however, highlight the significance of the indication and identification of content and presentation in a document. A more detailed discussion of these issues is presented in [DDMR90].

Further discussion on the rendering of tables into some physical representation can be found in Section 4.4.

4.3 Ontological Description: Functional

In this section we consider the functional view of the table. We will be concerned with two issues: first, how information is 'read' from a table; second, how cells in a table can be distinguished according to their function in providing data or providing an access route to the data.

Before we discuss these points, it is important to establish why some form of 'functional' view of the table is needed. Let's consider some very basic issues regarding the table and its use in a document. We can think of the *primary role* of the table as being to *encode and present* information pertinent to the discourse or the document. Assuming that the information is of a regular nature, we might ask *where is the information in the table?* In answering this question, we can state that some components of the table (i.e. some cells in the table) appear to behave in a terminating or completing role: the information in the table 'stops', or is completed, when we get to those cells. This class of cells is distinguished from the other cells which are used, in part, to arrive at instances of these terminal cells. These terminal cells, then, can be thought of as representing the completion of an instance of the regular information presented in the table. This being the case, a suitable term for their functional aspect is **data cells**. A contiguous area of data cells in a table is termed a **data area**.

Data cells are not simply the leaves of the categories (see Chapter 3) as some categories are wholly contained in the access areas of the table. However, in the natural view of the table, the data cells are all leaves of categories.

4.3.1 Reading Tables

How is a single cell arrived at? The table must in some way be navigated to get to the data cell in question. A description of the cells involved in this navigation, including ordering information, is called a reading of the table. How do we know

which cells to use to get to a particular cell? We use our knowledge of the domain in conjunction with the cell contents and the table idiom.

Perhaps the key issue regarding local search (see Section 3.1.1) and the functional description of the table is: how do we know which cells to target?

The straightforward approach to this problem is to observe the gross physical structure of tables in general. As we take the upper area and the left area of the table to be involved in the indexing of the information in the table, the remainder may be considered the target area. However, it is fruitful to consider the problem from another perspective: that of the information in the table and its use in the document.

If, for example, the document containing the table includes all terms found in some category, while one category is distinguished by containing only unseen or new information, then we might assume that that category is the data of the table. This follows from some notion of *new information* being presented by the table.[2] If, on the other hand, there is not a unique area of the table which has this property, how might we decide which to target on? For example, the following table might be found in a document mentioning `movies` and `directors`, but not instances of each:

(!4.5)

MOVIE	DIRECTOR
Star Wars	Lucas
THX1138	Lucas
A Room With A View	Ivory

If asked the question `who made THX1138`, we can answer `Lucas`. Conversely, if asked `what did Ivory make?` we can answer `A Room With A View`. We can also say that `Star Wars`, `THX1138` and `A Room With A View` are all names of movies. Adding our knowledge of table layout, we might infer that the main information in the table is the name of the director, as this appears on the right side of the table. So, in order to arrive at a functional description of this table stating that the cells below the `director` label are the data cells, we could consider the meaning of cell contents *or* some physical factors to do with relative location. In fact, in this case, the physical

[2] As the introduction of the data cell and data area suggested, the data cell may not necessarily be the new information presented by the table, but the last piece of information required to present the complete relation which in itself is new or unseen. For example, a table cross referencing phone numbers with names provides information in the association of the names and the numbers, not in the presentation of the numbers alone.

table — the matrix arrangement of the cells — allows our assumptions about the role of cells to provide a correct interpretation. In summary, we take the directors' names as being the data in this table and the other cells function as **access cells**, indexing these data cell.

If we extend this example as in **Table (4.6)**.

(!4.6)

MOVIE	DIRECTOR	DATE
Star Wars	Lucas	1977
THX1138	Lucas	1972
A Room With A View	Ivory	1984

We now have to decide whether or not the `director` names are still data cells. Perhaps in this case the problem is not so great as we can consider the table to contain information about particular movies giving a data area of two columns (`director` and `date`). However, in cases described by the following two examples, the ambiguity requires a lot more than simple layout knowledge to resolve. Compare **Table (4.7)** with **Table (4.8)**.

(!4.7)

Parameter A	Parameter B	Effect A
10	10	3
10	20	1

(!4.8)

Parameter A	Effect A	Effect B
10	10	3
5	20	1

In **Table (4.7)** the 'value' of `Parameter B` for a 'value' of 10 for `Parameter A` is not 20. Rather, the 'value' for `Effect A` for the `Parameter` setting of 10 and 20 respectively is `1`. In the second case, we are happy to revert back to the interpretation mirrored by the movies and directors example, giving 'values' for `Effect A` and `Effect B` for different setting of `Parameter A`. The above demonstrates the role of content understanding in determining the functional areas of the table, a task which, even in the simplest cases, is not fully specified by the physical nature of the table.

4.3.2 Functional Description

We suggest a simple view of the **function of the table**[3] as a representation of its exploitation by the reader. Again we concentrate on the local search which has a specific, unique data cell. We can think of the functional description as being in some way the natural view of the table. For example, in the example below, the dark shading indicates the data cells and the light shading indicates the access cells. The data cells are the targets of the local search: the goal of the search is to access and understand the contents of these cells.

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

Of course, it is possible to read any cell in a table (we might, for example, infer new information regarding a particular category by reading off cells thought to be structured below a label in a sub- super-type relationship — for example, **Table (1.4)** indicates that `Houston` is a `CITY`). The goal of local search, however, is to provide information about the table as characterised in the above example.

4.4 Ontological Description: Structure

By **structure**, we mean that aspect of the table which restricts how we navigate the cells in the course of a local search operation. This section, then, aims to consider:

- The notion of the Simple Table Relation: a structural representation which indicates which cells may be accessed from a particular position in the table.
- The range of physical phenomena which indicate structure, the factors contributing to their use and appearance, and whether or not there is any particular significance which we must associate with those physical patterns with respect to the other ontological components.

[3] In [DHQ95] a presentation is given of a different type of functional description. That type of description is more akin to the presentation of the semantics of the table in this thesis.

4.4.1 Structure In Tables

The above physical and functional components of the table model do not include any notion of **organisation**[4] between or of the cells. The first task, then, is to determine if there really is such a thing as structure in tables; that the apparent organisation of cells is not merely a meaningless typographic effect. There are two related ways in which we can demonstrate the existence of cell organisation:

1. Reading Paths.

2. Canonical Tables.

Of course, we could trivially demonstrate the existence of structure by randomly reorganising the cells in a table and then demonstrating that all meaning is lost as in the following.

[4]By organisation we mean the arrangement of cells in the table. The structure of the table is part of the meaningful organisation of the table along with juxtaposition and ordering of category elements.

(4.9)

	-14		PERCENT CHANGE
1990	CITY	-56%	
New York	2, 245	984	1996
-7	Houston	688	-30
Chicago	854	791	Los Angeles
983	568	261	-54
Philadelphia	503	431	MURDERS

However, the following two methods are used for discussion and the introduction of further important concepts.

Reading Paths

A **reading path** is, briefly, a path which the reader takes through the array of cells when using the table to locate or read a particular piece of information. The path is essentially a logical notion indicating a route through cells which are logically adjacent and not an indication of a navigation through the table via cells which are physically adjacent to one another. The notion of cells being logically adjacent is one which will be explored later and which will motivate the Simple Table Relation, the basic unit describing the structure of the table.

The following example (**Table (1.4)**) demonstrates a reading of a table with two reading paths; the first derives from accessing the data cell (containing the text 791) from the stub, and the second from accessing the data cell from the head.

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

The two paths could be represented informally as the content of the cells concatenated using a special character ($\frown$), bracketing may be used to indicate the paths: (CITY$\frown$Chicago) and (MURDERS$\frown$1996).

The information read may be a unique cell (as in the above example), interpreted by combining information found in the reading path accessing it, or relational, gathering together a number of cells which form a relation in the table (local or global search — [GWK93]). The fact that the reading paths are restricted to a particular set of cell sequences (as demonstrated below) is an indication that there is some form of structure in the table. In other words, for a particular table, the reader cannot move from one arbitrary cell to another and still either 'make sense' of the information they are reading, or understand what the document is saying. In **Table (1.4)** below, what sense could be made of a reading strategy which read, in some order, the cells highlighted?

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

More specifically, we view the manner in which the table is interrogated by the user as being a combination of a number of reading paths. For example, a standard matrix table will have at least two reading paths for accessing a single cell in the matrix: a vertical and a horizontal path. The set of cells in the reading paths, together with the target cell is termed a reading of a table.

Canonical Tables

A concept related to reading paths, and one which also offers insight into the structural nature of tables, is that of canonical tables ([DHQ95]). A canonical table is, *superficially*, the table reduced to a relation as might be found in a relational database.[5] Producing a canonical table from a particular table preserves information about the restricted connectivity of cells as described by the notion of reading paths while discarding the structural information, in particular the order of the cells, the organisation of the categories and so on.

[5]This seems to be the target of at least one system aimed at processing tables for information content: [LV92] which suggests that we can view a table '*as one relation of a relational data base*'.

There are a number of ways in which this notion might be made concrete. Perhaps the most extreme form can be illustrated by the following example which is a reduction of the table in Figure 4.1, page 74:

(4.10)

Feature groups represented by degree 3	(Exhaustive)	176
Feature groups represented by degree 3	(Exhaustive)	63
Feature groups represented by degree 3	(Covisible)	122
Feature groups represented by degree 3	(Covisible)	46

This is equivalent to the alternative arrangement presented in **Table (4.11)** below.

(4.11)

(Exhaustive)	176	Feature groups represented by degree 3
(Exhaustive)	63	Feature groups represented by degree 3
(Covisible)	122	Feature groups represented by degree 3
(Covisible)	46	Feature groups represented by degree 3

(Note the dependency between cells which appear more than once: 'Exhaustive' only ever appears when 'Feature groups represented by degree 3' is present and never with 'Covisible'. The concept of data dependency and its implications with respect to reading paths and categories will be discussed in Section 4.5.)

Such reductions preserve some aspect of the information presented by a table, yet clearly something is lost. The structure of a table is not simply relational (i.e. an un-ordered tuple) and bears information which is required if the table is to be understood. This information reflects something of the order in which elements are to be combined (from a compositional semantics viewpoint). The aspect of the structure which is maintained by the canonical reduction is that which indicates the sets of cells encountered when a reading is performed. In addition, the canonical table reduces the utility of the table. Relationships and values can no longer be compared with respect to the interaction between categories which may be reflected in the original design.

Structure is about groups of cells and the order in which they occur.

What Does Table Structure Do?

Given that tables have some form of structure, we must now consider the purpose of the structure. Why not simply have the canonical form of a table? The answer to this relates first to the functional component of the model (Section 4.3): the

structure indicates how a reading of a table is formed.Secondly it affects the semantic interpretation of the table (Section 4.5). Investigating the semantic utility of structure results in a number of discussion points:

- What does structure facilitate? By introducing and manipulating the structure of the table, what can the author get out of it?
- What does structure indicate (in terms of the relationship between the content of the cells)?

Firstly, structure facilitates physical economy (see Section A.1.2, page 237). The manner in which structure is rendered (described later in Section A.2) means that, for example, grouping cells with equivalent values together to form a single cell reduces the amount of space required by the table as a whole (see Section 4.4.1, 85 for an example). Secondly, the organisation of cells allows the author to juxtapose certain information bearing elements to imply certain concepts. For example, placing statistics for two different years next to each other allows the reader to compare the values and also permits the author to suggest some qualitative factor they see in the information. For example, in the following, comparing the number of seats or votes for a year between parties is straightforward.

(4.12)

Parties	1923		1924	
	Votes	Seats	Votes	Seats
Conservative	5,514,000	257	7,855,000	419
Liberal	4,265,000	158	2,985,000	40
Labour	4,358,000	192	5,482,000	151

However, comparing seats between years for an individual party might be better achieved by the obvious rearrangement:

(4.13)

Parties	Votes		Seats	
	1923	1924	1923	1924
Conservative	5,514,000	7,855,000	257	419
Liberal	4,265,000	2,985,000	158	40
Labour	4,358,000	5,482,000	192	151

The second discussion point is related more to the semantics holding between the content-bearing elements of the table (the cell contents). Organising the table physically helps to indicate where certain inter cell relationships exist. Note, however, that sometimes the organisation of cells is performed purely for reasons of economy or spatial restriction and doesn't indicate any form of semantic relationship.

Structure and table organisation may also be important in introducing and positioning redundant, or functionally redundant, categories. A **redundant category** is one which repeats information found elsewhere in the table. A **functionally redundant category** contains information which can be deduced from one or more domains — such as the arithmetic difference between values. In the following, the shaded domain is functionally redundant.

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

There is some evidence that redundancy is important to the human table reading task ([Wri82]). In addition, redundancy, and especially functional redundancy, can be used to highlight certain domains of information and thereby focus the reader's attention on a particular part of the table.

4.4.2 Organisation in the Reading Path: Hierarchy and Relation

The reading path connects cells which are related through the *meaningful* organisation of the table. Organisation occurs for one or more of the following reasons:

- information bearing:
  1. To manipulate the adjacency of domains or cell values.
  2. To indicate semantic relationships between cells.
- non-information bearing:
  1. To provide spatial economy, and other aesthetic reasons.

2. As a result of the restriction to two dimensions.

The task of extracting structure is to provide an abstraction of the table which maintains enough information to preserve meaningful relationships, while the non-information bearing effects are lost. The essence of the problem is in the restricted vocabulary of the physical table: patterns which indicate meaningful structural relationships are often also those resulting from non-information bearing effects. For example, **Table (4.14)**.

(!4.14)

Animal			
Cat	Monkey	Horse	Otter
(Male)			
15 yrs	8 yrs	10 yrs	5 yrs

The physical pattern presenting the `Animal` category is the same as that surrounding the common value `(Male)` and the values below: (`Animal`↷`Cat`↷`(Male)`↷`15 yrs`) (`Animal`↷`Monkey`↷`(Male)`↷`8 yrs`) (`Animal`↷`Horse`↷`(Male)`↷`10 yrs`) (`Animal`↷`Otter`↷`(Male)`↷`5 yrs`).

Tables are generally thought to have a hierarchical structure which provides for the encoding of some form of relational information (the local and global search distinction, see page 46). In order to lend clarity to the model being developed here we must consider representational mechanisms which are not only viable in abstract terms, but which also perform robustly with all possible layouts and domains. In a simple two-column table with labels above each column, the hierarchical relationships are clear to see. However, what of the relationship between the horizontally aligned values in the domains? A number of examples indicate circumstances when the relationships between those cells are to be read in a comparative manner (for example the votes per political party in **Table (4.13)**, page 87). However, in other cases, notably the matrix table (see page 32), the left hand column plays a more hierarchical role, indexing the data in the matrix. As the factors which distinguish the different semantic interpretations of the table, and, in a more complex model of structure, would disambiguate hierarchical from relational arcs, are often subtle and at the limit of automatic processing, we adopt a form of structural relationship between cells which is essentially hierarchical.

This approach is called the distributed approach to the Simple Table Relationship (STR). In other words, in the following example, we don't encode any relationship

between 1 and 2.

(!4.15)

	A	B
X	1	2

Only the relationship between A and 1, B and 2, X and 1 and X and 2 is encoded. This leads to the notion of a key index (similar to that concept in relational databases) which is required in simple tables where the stub is not distinguished by some form of indentation as in **Table (4.15)** (e.g. the Parties column in **Table (4.13)**) which we assume to be the left most column. Of course, this functional view of the table is not always appropriate and there may be more than one column of access cells on the left (e.g. **Table (4.8)**).

Above (page 84), the STR was introduced as the basic unit describing the reading paths in a table. Now that the extent to which the logical relationships found in the table are to be represented has been established we can informally introduce the STR. A reading path describes a set of cells encountered when reading a particular data cell in the table. For each reading path, the STR represents the transition from one cell to another that the reader follows during this navigation. These transitions may simply be encoded as a set of arcs. The Simple Table Relation represents the complete set of arcs for a particular table, and the set of reading paths may be reconstituted from the STR be following all the possible transitions as indicated by the arcs. For example, the arcs (A, B) and (B, C) describe the reading path A⌒B⌒C.

4.4.3 Restrictions To STR

As it stands, the Simple Table Relation is in fact too simple. The following contains reading paths A⌒X⌒α and B⌒X⌒β.

(!4.16)

A	B
X	
α	β

However, a simple duple based notation would provide arcs from A to X and X to β. Such a representation would result in the potential generation of the path A⌒X⌒β which is not intended by this table arrangement. It requires a certain amount of **restriction** to fully capture the reading paths of a table. Consequently, any pair of

cells defined in the STR is restricted by a set of cells which must appear in the path which the pair partially describe.

In this case, a simple indication of the STR encoded by cell pairs would allow for a path to be constructed from A to X to β: (A, X) and (X, β) imply A⌒X⌒β. We will write the appropriate restrictions thus:

$$\{(A,X)/[],(B,X)/[],(X,\alpha)/[A],(X,\beta)/[B]\}$$

4.4.4 Structure Orientation, Conjunction and Disjunction

In the above, there has been no mention of the orientation of the structure. For example,

(!4.17)

A	B	C
	D	E

allows us to consider (without further explanation) that indexing occurs from A to B and from A to D and then to the data cells from B to C and from D to E. The access part of the table has a horizontal component. Naturally, due to the hierarchical nature of the horizontal access structure we think of these cells as being used in conjunction with each other. In other cases, even when there is no spanning structure to indicate clearly this horizontal conjunctive effect, cells in access areas are still read in a horizontal group, as in the example repeated below.

(!4.18)

Parameter A	Parameter B	Effect A
10	10	3
10	20	1

There is, however, another arrangement which again requires content knowledge to fully understand, in which the access structure works in a **disjunctive** manner, as can be seen in **Table (4.19)**.

(4.19)

Square Cake Tin	Round Cake Tin	Flour	Sugar
5"	6"	4oz.	3oz.
10"	14"	8oz.	6oz.

Here, we are interested in the quantities for the ingredients for *either* a square tin *or* a round tin, not for both. If we index on the square tin, we ignore the round tin

entries and *visa versa*. However, such subtleties are not of particular importance as the two categories are effectively dependent on each other due to the fact that the relationship between them cannot vary.

Finally, there is the possibility of optional or exemplary material in the index structure which is not strictly necessary for indexing the data cells.

(4.20)

Quote	English-age	
<*old English quote*>	Old	
<*middle English quote*>	Middle	
<*modern English quote*>	Modern	

The quotes[6]on the left of **Table (4.20)** are only present for illustrative reasons and are not the main indexing category in the stub.

It might be useful to supply a structural model of this which mirrors the disjunction and allows the reading of the data cells to be accessed *either* by the first disjunctive index *or* by the second. However, in the interest of simplicity, a pragmatic solution might be to allow the access structure to follow the horizontal conjunctive analysis and infer a simple equivalence relationship of some sort between the value in the structure.

4.4.5 Index Orientation

Generally, the head is a vertical structure; the complexities of its organisation are arranged from the top down. However, in some cases the head may appear to be organised like the stub: the distribution of cells is horizontal and then indexes the data cells vertically, as in **Table (4.21)**.

(4.21)

Problem size ($N \times n$)	10×10	30×30	50×50
Search space (${}^{N}P_{n}$)	3.6×10^{6}	2.7×10^{32}	3.0×10^{64}
Mutation prob. (optimal)	0.05	0.09	0.10
Population size	100	100	100
Number of generations	30-50	Infinity	Infinity
Solution quality	Exact	—	—

[6]This table was glimpsed in a talk by Bill Teahan given at the University of Edinburgh in 1999. Appropriate material from the relevant age appeared however it was not noted in sufficient time!

In some cases, a mixture of horizontal and vertical 'labeling' of categories can be seen. Such cases offer extreme ambiguity when attempting to determine the structural relationships by automatic means. For example, the following example (**Table (4.22)**) has a category (`Number of pairs`) with numerical values, and a category `Cable type` also with numerical values. However, it takes a certain amount of knowledge to determine that the `Cable type` is horizontal and not vertical over a column of cells with alphabetic content.

(4.22)

Number of pairs	2			5		
Cable type	1	2	3	1	2	3
Thickness of sheath/bedding (nominal	0.9	0.9	0.9	1.2	1.2	1.2

4.4.6 Table Orientation

In general, tables are oriented with the access areas in either the left hand stub or the head. However, either by design or poor organisation, some tables appear to have contrary layouts. In the following example, it seems more natural to use the information in the right hand column for indexing the values in the left, rather than *vice versa*.

(4.23)

Story B - Correct Transcript	
Similarity	Cluster Keyword
0.306	China
0.296	Olympic Games
0.252	Olympic Games, Barcelona, 1992
0.244	Favored nation clause
0.212	Chinese Americans
0.212	Drug testing
0.211	Olympic Games, Atlanta, 1996
0.209	Intellectual property rights
0.195	Swimming
0.183	Athletes

4.4.7 Worked Examples

The following worked examples illustrate the Simple Table Relationship (STR) and also serve to illustrate a possible representation.

Barney's Table

(4.24)

Advisor	Weight
dynamic-mobility	1
capture-mobility	1
global-threat	1
eventual-mobility	1
promote-distance	1
eradicate	1
vital	1
material	1

The cells found in the above are as follows:

- {Advisor, Weight, dynamic-mobility, capture-mobility, global-threat, eventual-mobility, promote-distance, eradicate, vital, material, 1, 1, 1, 1, 1, 1, 1, 1}

and the STR is:

```
{
  (Advisor, dynamic-mobility)/[],  (Advisor, capture-mobility)/[],
  (Advisor, global-threat)/[],     (Advisor, eventual-mobility)/[],
  (Advisor, promote-distance)/[],  (Advisor, eradicate)/[],
  (Advisor, vital)/[],             (Advisor, material)/[],
  (Weight, 1)/[],                  (Weight, 1)/[],
  (Weight, 1)/[],                  (Weight, 1)/[],
  (Weight, 1)/[],                  (Weight, 1)/[],
  (Weight, 1)/[],                  (Weight, 1)/[],
  (dynamic-mobility, 1)/[],        (capture-mobility, 1)/[],
  (global-threat, 1)/[],           (eventual-mobility, 1)/[],
  (promote-distance, 1)/[],        (eradicate, 1)/[],
  (vital, 1)/[],                   (material, 1)/[],
}
```

Wang's Table

(4.25)

Year	Term	Mark					
		Assignments			Examinations		
		Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
	Spring	80	65	75	60	70	70
	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
	Spring	80	80	70	70	75	75
	Fall	75	70	65	60	80	70

The cells are as follows:

- { Year, Term, Mark, Assignments, Examinations, Grad, Ass1, Ass2, Ass3, Midterm, Final, 1991, Winter, Spring, Fall, 1992, Winter, Spring, Fall, 85, 80, 75, 60, 75, 75, 80, 65, 75, 60, 70, 70, 80, 85, 75, 55, 80, 75, 85, 80, 70, 70, 70, 75, 75, 80, 80, 70, 70, 75, 75, 75, 70, 65, 60, 80, 70}

and the STR is:

{
(Year, 1991)/[], (Year, 1992)/[], (Term, Winter) /[],
(Term, Spring)/[], (Term, Fall)/[], (Term, Winter)/[],
(Term, Spring)/[], (Term, Fall)/[], (1991, Winter)/[],
(1991, Spring)/[], (1991, Fall)/[], (1992, Winter)/[],
(1992, Spring)/[], (1992, Fall)/[], (Mark, Assignments)/[],
(Mark, Examinations)/[], (Mark, Grade)/[], (Assignments, Ass1)/[],
(Assignments, Ass2)/[], (Assignments, Ass3)/[], (Examinations, Midterm)/[],
(Examinations, Final)/[], (Winter, 85)/[], (Winter, 80)/[],
(Spring, 80)/[], ...
(Fall, 80)/[], ...
(Ass1, 85)/[], (Ass1, 80)/[], ...
(Ass2, 80)/[], ...
(Ass3, 75)/[], ...
(Midterm, 60)/[], ...
(Final, 75)/[], ...

(Grade, 75)/[], ...
(Grade, 70)/[], ...
}

NASA Table

This example was previously presented as Figure A.1, page 238.

(4.26)

Mission	Crew	United States						Russia/USSR		
		All			Shuttle					
		Flight	People	Trips	Flight	People	Trips	Flight	People	Trips
Through 1990		69	162/10	270	16	119/10	199/16	72	85/2	152/3
1991										
STS-37	5	70	165/11	275/17	39	122/11	204/17			
STS-39	7	71	170/11	282/17	40	127/11	211/17			
Soyus TM12	3							73	88/3	155/4

The STR is:

```
{
  (Mission, Through 1990)/ [],  (Mission, 1991)          / [],  (Mission, STS-37)          / [],
  (Mission, STS-39)         / [],  (Mission, Soyuz TM12) / [],  (Crew, Through 1990)       / [],
  (Crew, 1991)              / [],  (Crew, 5)             / [],  (Crew, 7)                  / [],
  (Crew, 3)                 / [],  (United States, All)  / [],  (United States, Shuttle)/ [],
  (All, Flight)             / [],  (All, People)         / [],  (All, Trips)               / [],
  (Shuttle, Flight)         / [],  (Shuttle, People)     / [],  (Shuttle, Trips)           / [],
  (Russia/USSR, Flight)     / [],  (Russia/USSR, People) / [],  (Russia/USSR, Trips)       / [],
  (Flight, 69)              / [],  (People, 162/10)      / [],  (Trips, 279)               / [],
  (Flight, 16)              / [],  (People, 119/10)      / [],  (Trips, 199/16)            / [],
  (Flight, 72)              / [],  (People, 85/2)        / [],  (Trips, 152/3)             / [],
  (Flight, 1991)            / [],  (People, 1991)        / [],  (Trips, 1991)              / [],
  (Flight, 1991)            / [],  (People, 1991)        / [],  (Trips, 1991)              / [],
  (Flight, 1991)            / [],  (People, 1991)        / [],  (Trips, 1991)              / [],
  (Flight, 70)              / [],  (People, 165/11)      / [],  (Trips, 275/19)            / [],
  (Flight, 39)              / [],  (People, 122/11)      / [],  (Trips, 204/17)            / [],
  (Flight, 71)              / [],  (People, 170/11)      / [],  (Trips, 282/19)            / [],
  (Flight, 40)              / [],  (People, 127/11)      / [],  (Trips, 211/17)            / [],
  (Flight, 73)              / [],  (People, 88/3)        / [],  (Trips, 155/4)             / [],
  (Through 1990, 69)        / [],  (STS-37, 5)           / [],  (STS-39, 7)                / [],
  (Soyuz TM12, 3)           / [],  (1991, STS-37)        / [],  (1991, STS-39)             / [],
  (1991, Soyuz TM12)        / [],
}
```

4.4.8 Summary

The above describes the factors pertaining to the organisation of cells in a table. Structure facilitates physical economy and juxtaposition of elements, and indicates semantic relationships. Cell organisation is essentially the grouping of cells. This can occur in a number of ways, and is the physical vocabulary of the table. Reading paths link cells in the order in which they are read. Reading paths indicate either a relationship between cell contents, the organisation of cells for juxtaposition or the imposition of the two dimensionality of the page on the structure of the information.

Given the above model of the structure and organisation of the table we want to be able to infer potential structure from the logical cells which make up the table. It is clear that it would be impossible to infer all and only the STR pairs in the table from the actual layout of the cells, i.e. without inspecting the content, functional attributes and so on as demonstrated in [HD97]. Consequently, inspection of the logical cells will only produce a set of hypotheses about the structure of the table.

The general strategy should be to take the patterns described in Section A.2 and consider the possible hypotheses generated. Observing the conditions in which over generation occurs will provide us with information about the constraints required and how these constraints ought to be implemented.

It should be noted that a reading is not the same as the *Simple Table Relation*. A reading can be constructed from the STR, together with information from the functional component of the model.

4.5 Ontological Description: Semantics

A semantic model of linguistic communication ultimately describes what the world has to be like for a statement to be true (possible world semantics). Of course, this should naturally be the goal of the semantic interpretation of information presented in tables. In many cases the table has a transparent relationship with some sentential form of the same information, and we can make a number of assumptions which will allow us to arrive at the linguistic semantic model. However, a model containing implicitly those assumptions would not fully model the potential complexities of the table. Concretely, these assumptions are generally about the 'missing' information 'between' cell contents on a reading path. For example, in a simple table assigning values to attributes, we might think of the missing linguistic information as being

the assertion: `the value of X is Y`. What is required is that the analysis system recognises that an assertion is being made. However, as will be discussed later, the number of possible relationships between cell contents is large and hard to determine computationally.[7]

The basic task of reconstructing the information in the table, then, can be thought of as requiring the identification of components of the table that need to be considered in combination; and the identification of the 'missing' material. This 'inter-cell' material can be cast as a *relationship* between (an interpretation) of the cell contents. This is the motivation for the following view of the semantic interpretation of tables.

We investigate the notion of a semantic interpretation of tables at a number of levels. Again, the robust nature of the proposed application and the incomplete resources available to domain independent language processing play an important role in the proposal of a suitable model. We aim, then, to cover the following topics, each of which contributes conceptually and systematically to our semantic view of the table:

- Relation Semantics: we propose that the table may be viewed as a relational information structure similar to database relations, plots, graphs and other non-linear document elements.

- Cell Content: the relation semantics advance a truth functional model of the table cells. However, the cells themselves are complex semantic entities and require further analysis. In addition, the analysis of certain cell contents, combined with the cell's functional role, influences the analysis and interpretation of other cell contents.

- Inter-Cell Relationships: relationships hold between the interpretation of the cell content elements in different cells.

- Organisational Semantics: the categories present may be ordered internally — siblings in a category are placed in order (e.g. alphabetically), and juxtaposed externally — categories are placed physically or logically next to each other in order to identify some comparative relationship.

[7] See discussion on the limit of table processing for information extraction in Section 1.4.

4.5.1 Semantic Components

As will be discussed in Section 4.5.4, the contents of cells may be either meta-text, object-text or a mixture of both. Generally, we are faced with a single span of object-text, or a single span of meta-text. However, in some cases we find a mixture of object and meta-text. The meta-text is used to indicate the position of the object-text and its relation with a superior cell. Consequently, it is possible for a cell to contain more than one component, each of which may participate in a different semantic relationship. To simplify matters, we will assume, in general, that the contents of a cell represent a unique semantic component, e.g. the semantic representation of a noun phrase. This will allow us to use references to cells formed from their content (i.e. quoted strings) when we are discussing them.

4.5.2 Relation Semantics

In this section we present the table as a relational structure and consider the semantic aspects of this interpretation. This view of the table is motivated by observing the similarities between the table and the relation, here characterised by the standard relational database model ([Ull88]. An equally important motivation is the requirement that there is some indication of where possible 'inter-cell relationships' exist. The use of a relational view of the table combined with the production of categories to describe the sets from which the relations take their values will provide a first step in this direction.

The Table as a Relation

For our purposes a **relation** is a function mapping the Cartesian product of a set of categories to true or false[8]. For example if category A contains $\{x, y\}$ and category B contains $\{1, 2\}$ the Cartesian product is $\{\langle x, 1\rangle, \langle x, 2\rangle, \langle y, 1\rangle, \langle y, 2\rangle\}$ and the relation $\mathcal{R}$ maps each element from this set to the boolean domain $\{true, false\}$ ($true$ indicating that the product represents a member of the relation). The relation acts as a filter removing elements from the Cartesian product. In practice all members will be present and so the relation will not be described or used further.

Viewing a table as a relation requires that we identify the categories which the Cartesian product operates on. We take a very simple view of the categories ap-

[8]Following the definition in [Wan96].

pearing in the relation and require that they only have members and not a 'label' or 'title' (other than such identifiers as might be used to catalogue the categories, rather than any identifying string which might come from the table itself) naming positions in the relation. This decision is motivated by observing that, often, what are clearly sets of similar values in a table are not in any way labeled by a superior cell, thus providing a label/value model of that set. For example, whereas in one table a category may have a root node or 'label' `CITY` and children identifying various cities (`New York`, etc.) other instances might simply list the cities. In the former case, we might want to create a category 'labeled' `CITY` with members as described. However, in the later case, there is no appropriate label.

Categories are mentioned in [Lef89], [GBB91b], and [Wan96], though only vague definitions are given.[9] This conceptual problem is due to the high-level organisation concepts which are being exploited by the author in order to guide the activity of data access. The representational structures needed if we were to attempt a formal definition are beyond current practical linguistic systems. Perhaps the best we can do is to reiterate Camerons's notion of 'categories of information' (page 34) which we may consider as having hierarchical structures which are some form of semantic tree sub-dividing the root semantic object according to a particular 'logical connection' (page 47).

The notion of the category under discussion here may be termed a **data category**, that is category information derived in terms of data dependency. This definition can then be used to provide a definition of a category based on the relationships holding between the data categories component parts.

The first task is to define what a data category is.

(!4.27)

	A
B	2

In the above (**Table (4.27)**), we would simply identify the relation which holds for members of the three categories {A}, {B} and {2}. There is no question as to how we read this table or which elements of the table are members of the same category (as they are all unique).

[9] The definitions are *vague* in that the cited publications do not indicate how to find the categories in a table, though they may indicate what to do with them once they are known, and, as in Wang's thesis, indicate precisely how they are captured notationally.

(!4.28)

	X	Y
A	1	2

In the case of **Table (4.28)** we must decide if 1 and 2 are of the same category and if X and Y are of the same category. If this is the case then we have a 3 category relation $\{\langle A, X, 1\rangle, \langle A, Y, 2\rangle\}$ — remember that we are omitting defining $\mathcal{R}$ for brevity. However, if we don't consider the above to be of the same categories then we have to consider two cases: the relation $\{\langle A, X, 1, Y, 2\rangle\}$ and the existence of 2 relations $\{\langle A, X, 1\rangle\}$ and $\{\langle A, Y, 2\rangle\}$. It is important to note that if X and Y are not in the same category, then nor are 1 and 2.

We can extend the factors which illustrate this decision point by extending the table as follows:

(!4.29)

	X	Y
A	1	2
B	3	4

Is this the relation over the set of categories $\{\{A, B\}, \{X, Y\}, \{1, 2, 3, 4\}\}$ or two relations over the set of categories $\{\{A\}, \{X, Y\}, \{1, 2\}\}$ and $\{\{B\}, \{X, Y\}, \{3, 4\}\}$, or $\{\{A, B\}, \{X\}, \{1, 3\}\}$ and $\{\{A, B\}, \{Y\}, \{2, 4\}\}$. The distinction between these representations is further illustrated by the following:

(!4.30)

	X	Y
A	1	2
P	α	β

In **Table (4.30)** the lexicographic distinctions lead us to interpret the table in the second manner ($\{\{A\}, \{X, Y\}, \{1, 2\}\}$ and $\{\{P\}, \{X, Y\}, \{\alpha, \beta\}\}$) as the first interpretation leads to rather inconsistent categories ($\{\{A, P\}, \{X, Y\}, \{1, 2, \alpha, \beta\}\}$). To complete the decision point description, we might consider the larger set $\{\{A\}, \{P\}, \{X, Y\}, \{1, 2\}, \{\alpha, \beta\}\}$.

The above decisions may appear arbitrary, however we may appeal to the notion of table reading as reported in the psychological literature ([GWK93], [Wan96], [WHL84]), to form some answers.

We may apply the notion of local and global search (Guthrie, [GWK93]) (introduced on page 47). Local search is the task of reading a single cell in the table, global search is the task of accessing (for the purpose of comparison etc.) a set of

cells in the table. We assume that the cells being accessed are those which appear in the data area rather then the access area of the table (in functional terms). We consider the task of global search to be equivalent to a set of local search operations. The global search has an objective which is related to the information domain of the document/table. Consequently, we assume that cells accessed by local search appearing in the same global search stand in some appropriate relationship to each other in terms of the reader's understanding of the information/domain of the table/document. In other words, there is a relationship between 1 and 2 and α and β in the above example.

The point in question is the interpretation of the local search: is it an evaluation to *true* for a relation, or, in some way, a subset of category values for a relation. If the second case is considered we must ask what is the subset of the set of category values which are distinguished by the data cell? If the first is considered, we must ask which are the other true points for that relation.

The solution proposed is that the local search is a *true* value for a relation, and that the global search is a larger construct which defines the table relation. This should be considered in conjunction with the functional view of the table. This issue was previously mentioned in the discussion regarding the Simple Table Relation on page 90.

The relational view we are constructing here is one which is used to build an interpretation of individual data cells, not one which is being used to build an interpretation of the data cells linked relationally in terms of the global search.

A Relational Model of the Table

This section advances a characterisation of the table as a relation as presented in Section 4.5.2. We wish to produce a bottom up, data oriented motivation for the category/relational view of the table to complement the top down definitions found in the literature (most notably [Wan96]). An example (**Table (4.31)**) is presented in parallel to illustrate and clarify the definition.

(4.31)

States		ϵ	b
q	sequence	q	qq
	probability	1.0	.2
r	sequence	r	qr
	probability	0.0	.1

We define a reading of a table to be a set of the cells encountered when accessing a target data cell in the table while performing a local search operation. It is the union of the cells found in the reading paths for the cell and the data cell itself. The reading set, or **table reading**, is the set of all such readings for a table. A data cell is a cell which appears only once in the reading set (another definition of the functional model of the table).

For **Table (4.31)**, the readings set is:

{
{States, q, sequence, ϵ, q},
{States, q, sequence, b, qq},
{States, q, probability, ϵ, 1.0},
{States, q, probability, b, .2},
{States, r, sequence, ϵ, r},
{States, r, sequence, b, qr},
{States, r, probability, ϵ, 0.0},
{States, r, probability, b, .1}
}

The **access set** is the reading set with the data cells removed.

{
{States, q, sequence, ϵ},
{States, q, sequence, b},
{States, q, probability, ϵ},
{States, q, probability, b},
{States, r, sequence, ϵ},
{States, r, sequence, b},
{States, r, probability, ϵ},
{States, r, probability, b},
}

Cells which only ever appear in a reading (from the access set) when other cells are present are said to be **dependent** on those cell contents.

As the data cell is the target of the search, we know that it will be dependent on all the cells in its reading. In addition, at this stage we consider cells to be the unique elements we are dealing with and do not include any model of identity among apparently equivalent cell contents.

The dependencies can be recorded as dependencies between the cells (i.e. a set of cell pairs called the **dependency set**). It must be noted (for reasons which will be made clear later) that these dependencies must also encode the number of times they appear in the reading paths.

$$\{\langle \text{States, q}\rangle_4, \langle \text{States, r}\rangle_4, \langle \text{States, sequence}\rangle_4,$$
$$\langle \text{States, probability}\rangle_4, \langle \text{States, } \epsilon\rangle_4, \langle \text{States, b}\rangle_4\}$$

Dependency is **transitive**, and we define the **maximal dependency set** to be a set and represent this as a set of dependent cells sets. For example, if the dependency set is $\{\langle cell_0, cell_1\rangle, \langle cell_1, cell_2\rangle\}$ then the maximal dependency sets (actually, only one in this case) are $\{\{cell_0, cell_1, cell_2\}\}$. These sets will later be rendered as cell sequences written using a dotted notation. The complete dotted representation can be thought of, for convenience, as a string which will ultimately name an element of a category.

A cell's contents can only appear in as many maximal dependency sets as the number of times it appears in the reading. This encodes the intuition/hypothesis that, in terms of compositional semantics, a cell's contents can't modify the contents of cells from *different* table domains. If a conflict is found when generating the maximal dependency set the dependency set may be modified to effectively filter out bogus dependencies (due to the numeric constraint outlined above) thereby resulting in singular dependency sets.

Calculating the maximal dependency sets for **Table (4.31)** requires that we make a decision about where the cell `States` is to go. This decision is motivated by the number of occurrences that the cell represents in the current dependencies set: in total 24. This is an issue of semantics. In this case, we suggest:

$$\{\{\text{States, q}\}, \{\text{States, r}\}, \{\text{sequence}\}, \{\text{probability}\}, \{\epsilon\}, \{\text{b}\}\}$$

Note that the dependencies have been filtered to account for the number of occurences of the cell containing the string **States** and that, as there are no possible transitive links in the resulting set, the maximal dependency set is smaller than the original set of dependencies.

Cell contents never appearing together in a reading are **mutually independent**. Maximal dependency sets which repeat are **category values**, and category values which contain mutually independent cells form categories. The following is an indication of the categories derived in this manner from **Table (4.31)**.

$$\{ \{\text{States.q, States.r}\}, \{\text{sequence, probability}\}, \{\epsilon, \text{b}\}\}$$

A table relation holds between readings (from the reading set) with one category fixed. If there are readings in the reading set which have no category in common, we call these readings **independent**, and the table is a **compound table**.

Worked Examples

The following worked examples provide more details illustrating the above definitions.

Worked Example 2

(4.32)

Mission	Crew	United States						Russia/USSR		
		All			Shuttle					
		Flight	People	Trips	Flight	People	Trips	Flight	People	Trips
STS-37	5	70	165/11	275/17	39	122/11	204/17			
Soyuz TM12	3							73	88/3	155/4

The reading set is:

{
{Mission, STS-37, Crew, 5, United States, All, Flight, 70},
{Mission, STS-37, Crew, 5, United States, All, People, 165/11},
{Mission, STS-37, Crew, 5, United States, All, Trips, 275/17},
{Mission, STS-37, Crew, 5, United States, Shuttle, Flight, 39},
{Mission, STS-37, Crew, 5, United States, Shuttle, People, 122/11},
{Mission, STS-37, Crew, 5, United States, Shuttle, Trips, 204/17},
{Mission, Soyuz TM12, Crew, 3, Russia/USSR, Flight, 73},

{Mission, Soyuz TM12, Crew, 3, Russia/USSR, People, 88/3},
{Mission, Soyuz TM12, Crew, 3, Russia/USSR, Trips, 155/4}
}

The access set is:

{
{Mission, STS-37, Crew, 5, United States, All, Flight},
{Mission, STS-37, Crew, 5, United States, All, People},
{Mission, STS-37, Crew, 5, United States, All, Trips},
{Mission, STS-37, Crew, 5, United States, Shuttle, Flight},
{Mission, STS-37, Crew, 5, United States, Shuttle, People},
{Mission, STS-37, Crew, 5, United States, Shuttle, Trips},
{Mission, Soyuz TM12, Crew, 3, Russia/USSR, Flight},
{Mission, Soyuz TM12, Crew, 3, Russia/USSR, People},
{Mission, Soyuz TM12, Crew, 3, Russia/USSR, Trips}

The dependent cell contents are:

{⟨Mission, STS-37⟩, ⟨Mission, Crew⟩, ⟨Mission, 5⟩, ⟨Mission, United States⟩,
⟨Mission, All⟩, ⟨Mission, Flight⟩, ⟨Mission, People⟩, ⟨Mission, Trips⟩,
⟨Mission, Shuttle⟩, ⟨Mission, Soyuz TM12⟩, ⟨Mission, 3⟩, ⟨Mission, Russia/USSR⟩,
⟨STS-37, 5⟩, ⟨STS-37, United States⟩, ⟨STS-37, All⟩, ⟨5, United States⟩,
⟨5, All⟩, ⟨United States, All⟩, ⟨United States, Shuttle⟩, ⟨Soyuz TM12, 3⟩,
⟨Soyuz TM12, Russia/USSR⟩, ⟨3, Russia/USSR⟩}

Maximal dependency requires that we decide about the dependency of 'Mission', 'STS-37', 'Crew', 'Soyuz TM12', '5', '3', 'United States', 'Russia/USSR', 'All', 'Shuttle'.

{{Mission, STS-37}, {Mission, Soyuz TM12}, {Crew, 5}, {Crew, 3},
{United States, All}, {Unites States, Shuttle}, {Russia/USSR}, {Flight},
{People}, {Trips}}

And the categories are as follows.

{{Mission.STS-37, Mission.Soyuz TM12}, {Crew.5, Crew.3}, {United States.All, United States.Shuttle, Russia/USSR}, {Flight, People, Trips}}

Worked Example 3

(4.25)

Year	Term	Mark					
		Assignments			Examinations		
		Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
	Spring	80	65	75	60	70	70
	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
	Spring	80	80	70	70	75	75
	Fall	75	70	65	60	80	70

The access set is:

{
{Year, 1991, Term, Winter, Mark, Assignments, Ass1},
{Year, 1991, Term, Winter, Mark, Assignments, Ass2},
{Year, 1991, Term, Winter, Mark, Assignments, Ass3},
{Year, 1991, Term, Winter, Mark, Examinations, Midterm},
{Year, 1991, Term, Winter, Mark, Examinations, Final},
{Year, 1991, Term, Winter, Mark, Grade},
{Year, 1991, Term, Spring, Mark, Assignments, Ass1},
{Year, 1991, Term, Spring, Mark, Assignments, Ass2},
{Year, 1991, Term, Spring, Mark, Assignments, Ass3},
{Year, 1991, Term, Spring, Mark, Examinations, Midterm},
{Year, 1991, Term, Spring, Mark, Examinations, Final},
{Year, 1991, Term, Spring, Mark, Grade},
{Year, 1991, Term, Fall, Mark, Assignments, Ass1},
{Year, 1991, Term, Fall, Mark, Assignments, Ass2},
{Year, 1991, Term, Fall, Mark, Assignments, Ass3},
{Year, 1991, Term, Fall, Mark, Examinations, Midterm},
{Year, 1991, Term, Fall, Mark, Examinations, Final},
{Year, 1991, Term, Fall, Mark, Grade},
{Year, 1992, Term, Winter, Mark, Assignments, Ass1},
{Year, 1992, Term, Winter, Mark, Assignments, Ass2},
{Year, 1992, Term, Winter, Mark, Assignments, Ass3},

{Year, 1992, Term, Winter, Mark, Examinations, Midterm},
{Year, 1992, Term, Winter, Mark, Examinations, Final},
{Year, 1992, Term, Winter, Mark, Grade},
{Year, 1992, Term, Spring, Mark, Assignments, Ass1},
{Year, 1992, Term, Spring, Mark, Assignments, Ass2},
{Year, 1992, Term, Spring, Mark, Assignments, Ass3},
{Year, 1992, Term, Spring, Mark, Examinations, Midterm},
{Year, 1992, Term, Spring, Mark, Examinations, Final},
{Year, 1992, Term, Spring, Mark, Grade},
{Year, 1992, Term, Fall, Mark, Assignments, Ass1},
{Year, 1992, Term, Fall, Mark, Assignments, Ass2},
{Year, 1992, Term, Fall, Mark, Assignments, Ass3},
{Year, 1992, Term, Fall, Mark, Examinations, Midterm},
{Year, 1992, Term, Fall, Mark, Examinations, Final},
{Year, 1992, Term, Fall, Mark, Grade}

The dependent cell contents are, after filtering for commitment due to semantic restraints:

{⟨Year, 1991⟩, ⟨Year, 1992⟩, ⟨Term, Winter⟩, ⟨Term, Spring⟩, ⟨Term, Fall⟩, ⟨Mark, Assignments⟩, ⟨Mark, Examinations⟩, ⟨Mark, Grade⟩, ⟨Assignments, Ass1⟩, ⟨Assignments, Ass2⟩, ⟨Assignments, Ass3⟩, ⟨Examinations, Midterm⟩, ⟨Examinations, Final⟩ }

The maximal dependency sets are:

{{Year, 1991}, {Year, 1992}, {Term, Winter}, {Term, Spring}, {Term, Fall}, {Mark, Assignments, Ass1}, {Mark, Assignments, Ass2}, {Mark, Assignments, Ass3}, {Mark, Examinations, Midterm}, {Mark, Examinations, Final}, {Mark, Grade} }

The categories are:

{{Year.1991, Year.1992}, {Term.Winter, Term.Spring, Term.Fall},
{Mark.Assignments.Ass1, Mark.Assignments.Ass2, Mark.Assignments.Ass3,
Mark.Examinations.Midterm,
Mark.Examinations.Final, Mark.Grade}}

which are the same as the categories presented in [Wan96].

Conjunction and Disjunction of Categories

So far, we have used the dot notation to indicate elements in a set called a category. Leading towards a more formal definition we now identify each of the dot separated components of those elements as being categories. The dot sequence represents a path down a tree. Consequently, we can think of categories on this path as being in a sub- super-category relationship.

Categories which are related by the sub- super-category relationship are obviously used in conjunction. However, there are other situations in which categories not thus related must still be used in conjunction. There are two major case in which this is observed. The first is that of horizontal and vertical reading paths. When a cell is read it generally is read using a horizontal path and a vertical path. Categories discovered via these two paths must clearly be used in conjunction. The informational reason for the recapitulated domain is the same as for the horizontal and vertical distinction and as such must simply be treated as another dimension — one which has incidentally been deformed through the rendering of the table. Consequently, categories generated from the interpretation of recapitulated domains are to be used in conjunction just as those on strict horizontal or vertical paths.

Categories which stand in a disjunctive relationship are precisely those which are never encountered together when a data cell is being read.

Below (Section 4.5.4), an account of the relationships between cell content elements is presented. In general, the semantic relationships between categories in a sub-super- category relationship are considered, however, just as valid are those between conjoined categories.

Equivalence between Conjoined and Sub-Categories

As described above, the categories are derived in a largely data driven manner. Though a reasonable intuition regarding categories and their internal structure would rely on a semantic analysis when confronted with indeterminacy (in the worked examples, those places where dependencies had to be filtered), it is not always a computationally reasonable approach from the data motivated point of view. Due to the consequences of the category definition and method of derivation, we must

allow the potential for semantic interpretation to persist not only in the internal structure of the category, but also between categories which are conjoined.

For example, generally, recapitulated categories are in some sense independent semantic categories. However, this is not a requirement for their identification. Categories which might be reasonably presented as a continuous hierarchical structure in the table may also be split at some level simply to provide a matrix table type of structure.

A Data Driven Approach to Relational Semantics

The main points of the discussion above are as follows.

- cell readings are determined via the structure of the table.
- categories are determined in strongly data driven manner.
- the derivation of category structure (recursive sub-categories) provides constraints on the semantic interpretation of cell contents.
- extra dimensionality in the table, reflected by the syntactic effect of recapitulated domains, is equally subject to similar semantic interpretation.

Additionally, it may be possible to argue some semantic implications of the structure of categories. For example, what is the difference between a depth 2 category and a depth 3 category? In a depth 3 category is it more likely that the relationship between levels 1 and 2 is a type of? or is there anything of this sort to be discussed? In fact you could claim that relationships derived from the text are generally too specialised to hold in categories of depth greater than 2. So any deeper might well be a more universal relationship such as type of etc.

4.5.3 Categories and Data Categories

The above has been concerned with the identification of categories from a data driven point of view. However, simply providing an account of the table's categories in such a manner is clearly application specific and doesn't achieve the desired generality. Consider the following examples.

(!4.33)

Year	
1999	2000

(!4.34)

Wakako	
1999	2000

(!4.35)

Wakako		Matt	
1999	2000	1999	2000

In the first, there is a clear semantic relationship between Year and 1999 and 2000. In the second example, the same relationship doesn't hold and a 'weaker' relationship exists which is presumably dependent on the context in which the table appears. The third example is effectively the same as the second, with the 'years' recapitulated.

The data category analysis as presented above for the first table would result in the category (year, (1999, 2000)). For the second, the similar category (wakako, (1999, 2000)) would result. However the third would provide two categories (wakako, matt) and (1999, 2000). For what should be obvious reasons of semantics, we would really like the second and third examples to have the same effective structure, i.e. that the second example produces two categories (wakako) and (1999, 2000).

The reason for the discrepancy is simply that the data categories are derived in a bottom up manner from the structural level with no semantic considerations. If we consider categories from the top down, i.e. from the creation of the table, we would have the 'year' category and the 'name' and then the manner in which the table was designed would determine which of the structures would be used and consequently which of the data categories would result.

We can see the relationship between these **semantic categories** as being the decision to 'combine' categories to produce data categories. This, in turn, allows us to consider the following.

What semantic relationships between categories allow them to be combined into data categories?

In fact, the internal structure of categories is the starting point to considering the more general question:

What does the category structure of the table tell us about the semantic relationships between the components of the table?

These issues will be used to motivate the discussion on inter-cell relationships.

Contiguous categories are particular to the stub and occur when a category structure contains cells which must be used in conjunction (e.g. **Table (4.36)**).

(4.36)

Available Nitrogen %		Time of Application	Days Until Incorporated
NH4	Organic	Date	Days
50	33	Nov-Feb	
25	33	Nov-Feb	>3
50	33	Mar-Apr	
25	33	Mar-Apr	>3
75	33	Apr-Jun	
25	33	Apr-Jun	>1
75	15	Jul-Aug	
25	15	Jul-Aug	>1
25	33	Sep-Oct	
15	33	Sep-Oct	1

4.5.4 The Interpretation of Cell Contents

In this section we consider the elements found in the cell which we consider to be the basic currency of the (linguistic) semantic interpretation of the table.

The text found in a cell can be considered to be of two types: Object level and Meta level.

Object Level Text Object level text is straight forward stuff. It denotes an element of the table domain, or describes a table domain and is generally a linguistic fragment which will eventually be used in a compositional interpretation together with other cell elements and linguistic and pragmatic elements from the text and domain knowledge.

For example:

(!4.37)

bike	
wheel	frame

Meta Level Text Meta level text explains something to the reader about how the contents of the cell are arranged, or what to expect in the cells related to the cell in question. For example, indicating that the type of things denoted by the cell contents may be of either one type of another. Alternatively, it may indicating that the content of a cell may be of a certain type and another type conjoined by a particular textual device.

For example:

(4.38)

Category/Number of spindles	High-end, 3

shows a conjunction, while (P29):

(4.39)

Value/Probability	0.8	prob=1/3

shows a disjunction. Note in this case the type of value has to be explicitly indicated. Note also that the same textual/linguistic device is used to indicate conjunction and disjunction!

Another function of meta level text is to indicate the fact that a particular data point is not present, for example the use of `N/A` to indicate that the contents would not be available or applicable. Also, titles, labels, captions and legends may all appear in apparent cells in a table. All are meta level text in that they tell us something about the table and are not part of the structured information contained *in* the table.

Generally, we can identify two types of meta level text:

1. **structurally scoped**: the interpretation of the meta level text applies to the interpretation of cell contents or cell content elements in cells restricted to those available through certain structural effects. At this stage we don't commit to restricting the scope to the reading for example.

2. **globally scoped**: the interpretation of the meta level text applies to the entire table, or a portion of the table not predicted simply by other aspects of the table model. For example, it may apply to all cells excepting other meta level cells.

If a cell pair only contains object level text then the relationship between those cells is simply the relationship between the contents. The category is termed simple. If, on the other hand, a cell contains meta level text mixed in with the object level text, then the cell is said to be complex; and the category is said to be complex.

4.5.5 Classifying Inter-Cell Relationships

Here, we look at the classification of **inter-cell relationships** (ICRs). An inter-cell relationship is that relationship which holds between the interpretation of cell

content elements. These relationships can be broadly characterised as either those which are akin to the sentence level semantic interpretation found in NLP systems and those which encode some relationship requiring interpretation of the discourse contained in the document as a whole (see Section 1.4, **Table (1.3)**).

In some cases it is possible to apply standard sentence level semantic interpretation to 'adjacent' cell content elements as if they were simply found next to each other in a sentence, as in the following example.

(4.40)

Depth of Knowledge/Competency Levels for		
All Students	IS Minor	IS Major

Here, we could expand to three noun phrases: Depth of Knowledge/Competency Levels for [a]ll students, Depth of Knowledge/Competency Levels for IS Minor, Depth of Knowledge/Competency Levels for IS Major.

In other cases, the relationship which holds between elements is something which has to be deduced either by an examination of the elements themselves, or by an examination of the document as a whole.

It seems appropriate that we characterise ICRs in a hierarchical, type based manner. Such a classification strategy derives partly from speculation regarding the nature of ICRs and partly from considerations of the proposed algorithmic strategy for identifying them. As we intend to use various information resources as well as exploiting various features of cell contents and cell content elements to identify the nature of relationships, we require an ICR representation that may indicate general as well as specific relationships so that a process may back-off in the face of lack of knowledge. Naturally, the correct construction of such a hierarchical categorisation requires an experimental base.

Note the distinction between the set of theoretical features which we use to classify an ICR by (for example our knowledge that a car is a vehicle is not something which is implicit in the strings 'car' and 'vehicle'), and the set of computable features given those strings (e.g. the distribution of alpha-numeric characters, or an estimation of the part of speech). However, it should be pointed out that although we expect similarities to be evident in computable features, we also rely on information provided by other aspects of the model (e.g. structural information) as well as from

some general knowledge sources.[10]

Naturally, as with any such categorisation, there are controversial decisions to be made. In fact, an attempt at providing such a classification in which both our view of the world as well as our expectations of a number of computable features regarding orthographic representations of the world, the richness of knowledge sources, and the capabilities of linguistic analysis techniques are considered is possibly an ambitious endeavour. However, it seems more appropriate to make such an attempt rather than to list a set of ICRs which would not provide a system which could exhibit any sort of graceful degradation.

The basic division of relationships between cell contents is one based on their role in some process of semantic interpretations. The first division identifies relationships between the semantic interpretation of cell contents. The second division collects those relationships between cell contents which are in a sense meta to the first division and as such should be considered prior to identification and use of the object level relationships.

This basic organisation is illustrated in the following list.

1. Ontological
   (a) Heterogeneous/Hierarchical
      i. Nominal Super-type [Car, Ford]
         A. Qualitative [Car, Red], the value of an attribute
      ii. Partitive [Car, Wheel]
      iii. Quantitative
         A. Units of Measure [Car, Per_Capita]: might be Nominal Super-types
         B. Quantitative Value [No. Cars, 2], the value of an attribute
      iv. Discourse (External Relationship)
2. Linguistic
   (a) Heterogeneous/Hierarchical
      i. Sentential

[10] Future work will require an investigation into the techiniques mentioned in [Hea98] which look at patterns in text which are indicitive of relationships between objects in the content domain.

A. Sentential Disjunction [Depth_of_knowledge, in_prolog]

(b) Homogeneous

i. Linguistic

A. Elliptic [number of people arrested, number of women]

The combination of the ontological and the linguistic relationships provides the full semantic analysis.

In a hierarchy, a superior node is in some way general to the nodes inferior to it. It expresses a concept which includes the concepts expressed below it in the hierarchy. The superior cell's contents express a concept which is specialised by the inferior cell. Hierarchical relationships can exist anywhere on the complete reading path, which may include fragments of table structure from orthogonal parts of the table.

In some context, the relationships between the superior cell and the inferior cell have an obvious sentential counterpart. For example, [City.New York] could be rendered as 'the City of New York'. However, it is often the case that this information would not ordinarily appear in a prose version of the same phrase and appears only to indicate the fact that a set is being identified. The extra knowledge (that New York is a city) is assumed common knowledge.

Additionally, the structure of the table can be thought of as replacing certain portions of text that would normally appear in a prose interpretation. [Person.john.sister] would be rendered as 'the person John's sister' where the relationship indicated by the structure represents, or may be replaced by, the possessive ''s'. Still in further cases, the structure only represents a linguistic disjunction (see the **Table (4.44)**).

In this light, the problem of interpreting the contents of the table can be reduced to the problem of figuring out which spans of text might be appropriately used to replace the structure within and between categories.

We can extend this analysis into sentence processing by suggesting that there is a certain level at which information is elided. We don't need to say 'The person John', or 'The physical animate agent that is a person that is John'. This information is considered to be part of the pragmatic and reasoning or knowledge component of a system. As mentioned above, some of the information found in the table would not be present in the prose rendering and so we can infer that that additional information

is essentially knowledge that is to be exploited if known, or to be remembered and learned.

Ontological Relationships

Nominal Super-type Possibly the simplest and most common relationship found. A noun phrase in the superior cell denotes the type of things denoted by the noun phrase in the inferior cell.

For example **Table (4.41)**:

(4.41)

CITY
New York
Los Angeles
Chicago
Houston
Philadelphia

- Qualitative Relationships

  The noun phrase in the superior cell denotes an entity which has features, attributes or qualities denoted by the noun phrases in the inferior cells.

Partitive The noun phrase in the superior cell denotes an object composed of parts denoted by the noun phrase in the inferior cells.

Quantitative Relationships

- Units of Measure

  The inferior cell indicates by which unit of measure a superior cell is quantified.

  For example (P8):

(4.42)

ZD Business Disk WinMark 97
Thousands of bytes/sec

- Quantitative Value

  The inferior cell denotes a value of the type described by the superior cell.

  For example (P15):

(4.43)

Percent of Total
11.83
16.94

Linguistic Relationships

Sentential Disjunction A sentence has a number of completions in inferior cells.

For example (P30):

(4.44)

Depth of Knowledge/Competency Levels for		
All Students	IS Minor	IS Major

External Relationships The relationship between the superior and inferior cells is explained in the textual content of the document.

Homogeneous Relationships

In order to provide a complete interpretation of some cells, it is sometimes necessary to find information in sibling cells which may provide some linguistic context, such as the case with elliptical cell contents.

4.5.6 An Analogy with Linguistics

In computational linguistics, the conventional way in which semantic interpretation is carried out is as follows:

1. Words are identified through lexical, morphological or stochastic processes, and a semantic description is associated with them together with the syntactic information.

2. The structure of the sentence is computed.

3. The semantic model is built up combining all the components using a single combinatorial operator.

4. Scoping and other issues are dealt with.

The semantic interpretation exists either as an enrichment of a world model represented by simple logical statements, or (in the case of IE) as a set of inter-linked templates (see Section 1.3).

In the case of the table, the homogeneous structure is used to complete cell elements (e.g. through the discharge of ellipsis) and cell internal interpretations through the use of the above linguistic relationships. Then the heterogeneous relationships are used to build up interpretations of the data items.

In the case of the table, the manner in which the heterogeneous relationships are selected is important. In the sentence case, once the syntactic pattern has been confirmed, the semantics follow via, for example, the application of the lambda operator. However, structure in the table requires the identification of the 'operators' — the relationship type — between components before a compositional interpretation can be performed.

Whereas in computational linguistics we can rely on the nature of the local semantic interpretation of components to be combined according to the simple application of an operator, in tables, there is no such analogy with the selection of an operator which is predictable from the structure of the table as presented here. The selection of the relationship between cell components is the selection of an operator to be used to combine the semantics of the components. In this section we suggest that there is a common set of relationships or operators to be used, however, we don't suggest that this is a closed class - rather the aim is to provide some useful point to start. The description of relationships allows for 'external relationship' (effectively anything which can be assumed from knowledge of the domain, or which may be described using language in the body of the document, or some part of the table containing meta-text).

The relationships between categories may be found through standard linguistic processes. However, it is also the case that the relationships may be complex, amounting to missing information which may be sought in domain knowledge or in the document

as a whole and which may be applied to the combination of cell element interpretations (i.e. as an operator between cell elements) and/or which may be applied to cell elements in order to provide their interpretation.

4.5.7 A Comparison with Theories of Context Dependence

Interpreting and understanding a table, once categories have been established, can be characterised as 'finding missing information'. This information can be discovered in the document itself, in models of the domain which the document's content derives from or is about, or from world knowledge. This task bears some similarity to the interpretation of human language containing ellipsis.

There is much work in this area, however research into formal and computational aspects of the problem is possibly best described by the work carried out at SRI Cambridge ([LP95], [Pul94], [Cro95]).

The basic task in the sentential context is to realise the relationship between material which has gone before and material which is in some sense missing from the current statement. For example, having said `Wakako enjoyed the movie`, we can interpret `So did Matt`, by identifying the lack of verb in the later sentence and the appropriate material in the former.

In some cases, this relationship between statements with missing parts and prior material can be found between cell elements themselves as in the following, between `No Of People Stopped For Importation` and `No Of Women`.

(4.45)

Substance	No Of People Stopped For Importation	No Of Women
Synthetic drugs (Ecstasy/amphetamins/LSD)	248	28
Herbal cannabis	905	135
Cannabis resin	1190	134
Cocaine	311	102
Heroin	124	20

In the more general case, we are faced with having to establish when relationships must be sought elsewhere in the text, domain or world knowledge and precicely where this might be. For information found in the document containing the table, we need to establish where to search. For this, a model of the manner in which tables are

discussed is required in order to classify sentences as being candidates for providing 'missing' information. In addition, the system would have to be sensitive to detecting and processing both the explicit introduction of terms and their relationships as well as the implicit.

(4.46)

MOVIE	1st WEEKEND BOX OFFICE SALES	No OF SCREENS	PER SCREEN AVERAGE
The Lost World: Jurrasic Park	$92,729,064	3,218	$28,262
Mission: Impossible	56,811,602	3,012	18,862
Batman Forever	52,784,433	2,842	18,573
Independence Day	50,228,264	2,882	17,478
Jurassic Park	50,159,460	2,404	20,865

There are other examples of context dependent cell contents in the data area of the table. In this example (**Table (4.46)**), information found in one data cell is implied for the others below it. The currency symbol $ is found only in the top cell in the column.[11]

This also happens in the next example (**Table (4.47)**) where the counter (`images`) is only mentioned in the first row of data cells.

(4.47)

Image Quality Mode	C-1400L*	C-1000L*
SHQ	4 images	6 images
HQ	12	20
SQ	49	49

Finally, there are examples in which the meaning of cells is distributed through certain neighbouring cells via the fracturing of the components of a linguistic whole. In the following example (**Table (4.48)**), the text in the second access category (`Description`) 'runs through' a number of cells: `sublanguage, cache` → `and weighted` → `cache model`. An interpretation of the contents of these cells would require that we recognise the cells are related and that we can string them together to develop the overall meaning of each cell.

[11] This phenomenon also occurs in list structures, for example the list in Section 8.2.1.

(4.48)

Site (Year)	Description	Result	Ref.
IBM (91)	cache model		[2]
CMU (94)	trigger model	19.9→17.8	[3]
BU (93-94)	clustering	11.3→11.2	[4]
NYU (94-96)	sublanguage, cache	11.0→10.6	[5]
	and weighted	24.6→24.0	[6]
	cache model	33.3→33.0	
CMU (96)	hand clustering	0.1,0.6% improve.	[7]
SRI (96)	clustering	33.1→33.0	[8]
CU (96)	cache model	27.7→27.5	[9]

4.5.8 Combining Relational Semantics and ICRs

The basic architecture of the semantics of the table contains the relational view of the table and the inter-cell relationships described above. The categories derived for the relational semantics indicate where inter-cell relationships should be computed. The final step is to combine the interpretation of these categories with the others in the reading paths, and with the interpretation of the data cell contents which are the target of the individual readings.

In the majority of cases, an analysis which uses the same approach as for analysing the relationships between categories could be used. For example, a data cell containing a number will have obvious connections with a dominating access cell which indicates some unit of measure (e.g. `height in cm` → `170`). However, in other cases it might be more appropriate to provide an interpretation of the data cell as being some combination of a subset of the categories indexing it (particularly when the data cells are in an implied category). In addition, there may be some complex (e.g. functional) relationships between the indexing catgories which provide an interpretation of the cell (e.g. `p=10.q=5`→`17`).

Consequently, the interpretation of the data cell with resepct to the categories is something which needs consideration. One possibility is that if there is not an obvious relationship between the data cell and one of the categories (e.g. unit of measure and a numerical value), then any relationship should be considered as holding between all of the categories in some manner.

Data and Access Categories

After producing the categories of a table, we can identify data and access categories. A data category is one which describes the value in the cell being read.

Simple Conjunctive Interpretation

The simplest case for combining the interpretation of categories with the value being read can be thought of as expressing some form of condition: *when* X, Y *and* Z *then* V. For example, in the following

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

we might say *when the* CITY *is* New York *and the 'year' is* 1990 *then the 'number of'* MURDERS *is* 2, 245. The 'extra' information required for this interpretation can generally be deduced from the *type* of the cell elements: 2, 245 is a number, therefore 'number of', 1990 is a year, therefore 'year'.

Context Interpretation

When the *type* of the cell elements cannot be used to discover the relationships between the cell elements as described by a category, then the interpretation is context dependent, meaning that the document, domain or other world knowledge is required for a complete interpretation.

4.5.9 Meaning of Table Organisation

For now, we don't explore this area, but mark that we would like to have some indication of the meaning of ordering in a category and the significance of juxtaposition among categories.

4.5.10 Summary

This section has discussed the issue of semantics with respect to table understanding. A number of complementary semantic views of the table were proposed:

1. Relation Semantics.
2. Cell content semantics.
3. Inter Cell Relations.
4. Organisational Semantics.

Relation semantics provide an interpretation of the table similar to that of the database table. It is also proposed that the relation semantic view of the table may provide some insight into the inter cell semantic view of the table, though this issue has to be explored further. The notion of a category, as proposed by this relation semantic view is similar to that suggested elsewhere. The difference here is that we propose a strictly data dependent view: further investigation is required to determine how this compares to any intuitive understanding of categories in the table.

The cell content semantics discussion identified the need to distinguish components of a cell which have a meta level interpretation and those which have an object level interpretation. Inter cell relations are those meaningful relationships holding between the object level elements of cells. An initial typology of these relationships was presented.

4.6 Chapter Summary

This chapter has motivated and discussed the main components of a model of tables for linguistic applications. The table breaks down into the following four components.

1. Physical
2. Functional
3. Structural
4. Semantic

All of which can be used to define the space of possible tables and which provide parameters for the possible ambiguities in tables and the interaction between model components with respect to those ambiguities.

Chapter 5

The Model Representation

This chapter provides a representation of the table model. The purpose of providing this notational description is two fold. Firstly it provides a formal description of the class of document elements which we call tables. Secondly, it provides a language for describing algorithms designed for implementation in a table processing system which instantiates an instance of the model and exploits features of that instance.

5.1 The Table

We start the definition with a simple abstract encoding of the table.

definition:
A basic table, T, is a set C of cell identifiers.

5.2 Representation: The Physical Table

For our purposes, the table is represented physically as a number of cells. A cell has relative coordinates representing the top left and bottom right location of the cell. The contents of the cells are represented by strings. For the purposes of this definition, we don't characterise strings. It is sufficient to say that any processes of normalisation are consistent and applied to all strings. In general, for purposes of equality, it is assumed that the strings have unit spaces, and no other form of white space character (e.g. carriage returns or tabs). Conventionally, when describing

components of tuples describing the table representation, the dot ('.') is used to indicate the component much, as in programming conventions for record or object structure.

definition:
A cell, C, is a 6-tuple $\langle id, x_1, y_1, x_2, y_2, string \rangle$ *where id is the cell identifier* ($id \in T$), x_1 *and* y_1 *are the upper-left coordinates,* x_2 *and* y_2 *are the lower-right coordinates and the string is the cell contents.*

The table is defined in terms of these cells.

definition:
The **physical table,** T^{phys}, *is, then, a set of cells described in the above manner:* $\{\langle id_0, x_1^0, y_1^0, x_2^0, y_2^0, string_0 \rangle, \ldots, \langle id_n, x_1^n, y_1^n, x_2^n, y_2^n, string_n \rangle\}$
The following constraints are required to complete the definition.

- *no intersection of cells:* $\forall C \in T^{phys}, \neg\exists C' \in T^{phys} : (C = C') \vee$

$$(((C'.x_1 \leq C.x_2) \wedge (C'.x_1 \geq C.x_1)) \vee ((C'.y_2 \leq C.y_2) \wedge (C'.y_2 \geq C.y_1))) \wedge$$
$$(((C'.x_1 \leq C.x_2) \wedge (C'.x_1 \geq C.x_1)) \vee ((C'.y_1 \leq C.y_2) \wedge (C'.y_1 \geq C.y_1))) \wedge$$
$$(((C'.x_2 \geq C.x_1) \wedge (C'.x_2 \leq C.x_2)) \vee ((C'.y_1 \leq C.y_2) \wedge (C'.y_1 \geq C.y_1))) \wedge$$
$$(((C'.x_2 \geq C.x_1) \wedge (C'.x_2 \leq C.x_2)) \vee ((C'.y_2 \geq C.y_1) \wedge (C'.y_2 \geq C.y_2))).$$

- *no strings are empty:* $\neg\exists C \in T^{phys} : C.string = ""$.

We can define the following concepts based on the above definition of the physical table.

1. unary:
   (a) top size: the size of the top face of cell X ($\uparrow X \uparrow$) is the number of cells adjoining X on its upper face.
   (b) bottom size: the size of the bottom face of cell X ($\downarrow X \downarrow$) is the number of cells adjoining X on its lower face.

(c) right size: the size of the right face of cell X ($\vec{X}$) is the number of cells adjoining X on its right face.

(d) left size: the size of the left face of cell X ($\overleftarrow{X}$) is the number of cells adjoining X on its left face.

2. binary:

(a) neighbour access: the n^{th} neighbour of cell X on a particular face ($X[_{\{top,bottom,left,right\}}n]$) is accessed if present. Neighbours are numbered from left to right and from top to bottom.

(b) below: cell X is below cell Y ($X \downarrow Y$) if $X.y_1 > Y.y_2$.

(c) above: cell X is above cell Y ($X \uparrow Y$) if $X.y_2 < Y.y_1$.

(d) left of: cell X is left of cell Y ($X \leftarrow Y$) if $X.x_2 < Y.x_1$.

(e) right of: cell X is right of cell Y ($X \rightarrow Y$) if $X.x_1 > Y.x_2$.

(f) content equality: cell X has equal contents to cell Y if $X.string = Y.string$.

(g) spans horizontally: cell X spans horizontally cell Y ($X \cap Y$) if $((X.x_1 < Y.x_1) \wedge (X.x_2 \geq Y.ax_2)) \vee ((X.x_1 \leq Y.x_1) \wedge (X.x_2 > Y.x_2))$;

(h) spans vertically: cell X spans vertically cell Y ($X \subset Y$) if $((X.y_1 < Y.y_1) \wedge (X.y_2 \geq Y.y_2)) \vee ((X.y_1 \leq Y.y_1) \wedge (X.y_2 > Y.y_2))$;

(i) perfect align horizontal: cell X and cell Y are perfectly aligned horizontally ($X \Leftrightarrow Y$) if $(X.y_1 = Y.y_1) \wedge (X.y_2 = Y.y_2)$.

(j) align horizontal: cell X and cell Y are aligned horizontally ($X \leftrightarrow Y$) if $(X \Leftrightarrow Y) \vee (X \subset Y)$.

(k) perfect align vertical: cell X and cell Y are perfectly aligned vertically ($X \Updownarrow Y$) if $(X.x_1 = Y.x_1) \wedge (X.x_2 = Y.x_2)$.

(l) align vertical: cell X and cell Y are aligned vertically ($X \updownarrow Y$) if ($X \Updownarrow Y) \vee (X \cap Y)$

(m) adjacent: cell X is adjacent to cell Y ($X \bowtie Y$) if $((X.x_1 = Y.x_2+1) \wedge (X \leftrightarrow Y)) \vee ((X.y_1 = Y.y_2 + 1) \wedge (X \updownarrow Y)) \vee Y \bowtie X$.

(n) left margin: cell X is in the left margin if $\neg\exists C \in Tab : C.x_1 < X.x_1$.

(o) right margin: cell X is in the right margin if $\neg\exists C \in Tab : C.x_2 > X.x_2$.

(p) top margin: cell X is in the top margin if $\neg\exists C \in Tab : C.y_1 < X.y_1$.

(q) bottom margin: cell X is in the bottom margin if $\neg\exists C \in Tab : C.y_2 > X.y_2$.

Table (1.4) is used to illustrate the representation and some of the concepts defined above. Note that in this example, as the strings appearing in the cells also appear in the representation of the physical table, no effort is made to explicitly label the cells with unique identifiers in the illustrative table. In later examples where the physical table is not given, the cell contents are subscripted with a unique identifier so that they can be indexed in the representation.

(1.4)

	MURDERS		PERCENT CHANGE
CITY	1990	1996	
New York	2, 245	984	-56%
Los Angeles	983	688	-30
Chicago	854	791	-7
Houston	568	261	-54
Philadelphia	503	431	-14

The physical table, T^{phys} is represented as follows.

$$
\begin{aligned}
T^{phys} = \quad \{ & \\
& \langle cell_0, 1, 0, 2, 0, \texttt{MURDERS} \rangle, && \langle cell_1, 3, 0, 3, 1, \texttt{PERCENT CHANGE} \rangle, \\
& \langle cell_2, 0, 1, 0, 1, \texttt{CITY} \rangle, && \langle cell_3, 1, 1, 1, 1, 1990 \rangle, \\
& \langle cell_4, 2, 1, 2, 1, 1996 \rangle, && \langle cell_5, 0, 2, 0, 2, \texttt{New York} \rangle, \\
& \langle cell_6, 1, 2, 1, 2, 2, 245 \rangle, && \langle cell_7, 2, 2, 2, 2, 984 \rangle, \\
& \langle cell_8, 3, 2, 3, 2, -56\% \rangle, && \langle cell_9, 0, 3, 0, 3, \texttt{Los Angeles} \rangle, \\
& \langle cell_{10}, 1, 3, 1, 3, 983 \rangle, && \langle cell_{11}, 2, 3, 2, 3, 688 \rangle, \\
& \langle cell_{12}, 3, 3, 3, 3, -30 \rangle, && \langle cell_{13}, 0, 4, 0, 4, \texttt{Chicago} \rangle, \\
& \langle cell_{14}, 1, 4, 1, 4, 854 \rangle, && \langle cell_{15}, 2, 4, 2, 4, 791 \rangle, \\
& \langle cell_{16}, 3, 4, 3, 4, -7 \rangle, && \langle cell_{17}, 0, 5, 0, 5, \texttt{Houston} \rangle, \\
& \langle cell_{18}, 1, 5, 1, 5, 568 \rangle, && \langle cell_{19}, 2, 5, 2, 5, 261 \rangle, \\
& \langle cell_{20}, 3, 5, 3, 5, -54 \rangle, && \langle cell_{21}, 0, 6, 0, 6, \texttt{Philadelphia} \rangle, \\
& \langle cell_{22}, 1, 6, 1, 6, 503 \rangle, && \langle cell_{23}, 2, 6, 2, 6, 431 \rangle,
\end{aligned}
$$

$\langle cell_{24}, 3, 6, 3, 6, -14 \rangle$

}

The following are examples of statements which are true about T^{phys}.

- $\uparrow cell_3 \uparrow = 1$.
- $\downarrow cell_0 \downarrow = 2$.
- $cell_0[_{bottom} 1] = cell_4$.
- $cell_0 \cap cell_3$, $cell_0 \cap cell_4$.
- $cell_3 \Updownarrow cell_6$.
- $cell_1 \bowtie cell_0$

For convenience, in the following examples, the cell identifier is placed in the cell as a subscript to the string.

5.3 Representation: Functional

Functional information is a mapping from the set of cells to the domain $\{ACCESS, DATA\}$, and may be represented by set membership.

definition:
The functional table, T^{func}, is a tuple $\langle \mathcal{A}, \mathcal{D} \rangle$, where $\mathcal{A}$ is the set of access cell identifiers and $\mathcal{D}$ is the set of data cell identifiers.

1. $\forall X \in \mathcal{A}, X \in T.C$.
2. $\forall X \in \mathcal{D}, X \in T.C$.
3. $\mathcal{A} \cup \mathcal{D} = T.C$.
4. *cell X is an $ACCESS$ cell if $X \in \mathcal{A}$.*
5. *cell X is a $DATA$ cell if $X \in \mathcal{D}$.*

Naturally,

- $\mathcal{A} \cap \mathcal{D}$ is empty.

To illustrate the definition, the following table is presented.

(1.4)

	MURDERS$_{cell_0}$		PERCENT CHANGE$_{cell_1}$
CITY$_{cell_2}$	1990$_{cell_3}$	1996$_{cell_4}$	
New York$_{cell_5}$	2, 245$_{cell_6}$	984$_{cell_7}$	-56%$_{cell_8}$
Los Angeles$_{cell_9}$	983$_{cell_{10}}$	688$_{cell_{11}}$	-30$_{cell_{12}}$
Chicago$_{cell_{13}}$	854$_{cell_{14}}$	791$_{cell_{15}}$	-7 $_{cell_{16}}$
Houston$_{cell_{17}}$	568$_{cell_{18}}$	261$_{cell_{19}}$	-54$_{cell_{20}}$
Philadelphia$_{cell_{21}}$	503$_{cell_{22}}$	431$_{cell_{23}}$	-14$_{cell_{24}}$

The sets $\mathcal{A}$ and $\mathcal{D}$ defining the above table, comprising the functional table T^{func}, are as follows.

$$\mathcal{A} = \{cell_0, cell_1, cell_2, cell_3, cell_4, cell_5, cell_9, cell_{13}, cell_{17}, cell_{21}\}.$$
$$\mathcal{D} = \{cell_6, cell_7, cell_8, cell_{10}, cell_{11}, cell_{12}, cell_{14}, cell_{15}, cell_{16}, cell_{18}, cell_{19}, cell_{20}, cell_{22}, cell_{23}, cell_{24}\}.$$

5.4 Representation: The Simple Table Relation

The Simple Table Relation (STR) is an encoding of the reading paths in a table, as discussed in Section 4.4. If the relation holds between two cells then those cells are adjacent in a reading path. The tuples may be modified to indicate restriction in their usage determined by the presence of other cells in the reading.

definition:
The simple table relation (STR) is a set of triples, $\langle X, Y, R \rangle$, *where* X *is a cell identifier representing the source and* Y *is a cell identifier representing the sink of a directed arc and* R *a set of* n *restrictions* (r_0 *to* r_n) *on the transitition of the arc:* $\langle id_0, id_1, \{id_{r_0}, \ldots, id_{r_n}\}\rangle$. *The structural table,* T^{struc}, *is, then, described as a set thus:* $\{\langle id_0^0, id_1^0, \{id_{r_0}^0, \ldots, id_{r_n}^0\}\rangle, \ldots, \langle id_0^n, id_1^n, \{id_{r_0}^n, \ldots, id_{r_n}^n\}\rangle\}$

The following constraints are required to complete the definition.

- the relation must hold between two different cells: $\forall C \in T^{struc} : \neg C.X = C.Y$.
- any pair of cells can appear only once in the relation: $\forall C \in T^{struc} : \neg \exists C' \in T^{struc} : C.X = C'.Y \wedge C.Y = C'.X$.

We can define the following concepts based on the above structural definition of the table (T represents the basic table, i.e. the set of identifiers of the cells).

1. cell X immediately dominates cell Y ($X \succ Y$) if $< X, Y, R > \in T^{struc}$.
2. cell X dominates cell Y ($X \succ *Y$) if $X \succ Y \vee \exists A \in T : X \succ A \wedge A \succ *Y$.
3. $\mid P \mid_{\Downarrow}$ is the set of cells immediately dominated structurally by cell P.
4. $\mid P \mid^{*}_{\Downarrow}$ is the set of cells dominated structurally by cell P
5. $\overrightarrow{X}$ is the set of paths to X.
6. a set of cells P, a path to cell X, is defined recursively:
   (a) $\exists Y \in T : \langle Y, X, R \rangle \in T^{struc}, \exists P' \in \overrightarrow{Y}$, then, $P = \{Y\} \cup P' \wedge \forall C \in R : C \in P$.
   (b) $\neg \exists Y \in T : \langle Y, X, R \rangle \in T^{struc}, then, P = \{\}$.
7. a reading of a cell is the set of paths to that cell.

The following additional constraint is required:

- $\forall P \in \overrightarrow{X}, P$ appears exactly once.

Table **Table (1.4)** is used to illustrate the definition.

(1.4)

	MURDERS_{cell_0}		$\text{PERCENT CHANGE}_{cell_1}$
CITY_{cell_2}	1990_{cell_3}	1996_{cell_4}	
New York_{cell_5}	$2,245_{cell_6}$	984_{cell_7}	$-56\%_{cell_8}$
$\text{Los Angeles}_{cell_9}$	$983_{cell_{10}}$	$688_{cell_{11}}$	$-30_{cell_{12}}$
$\text{Chicago}_{cell_{13}}$	$854_{cell_{14}}$	$791_{cell_{15}}$	$-7_{cell_{16}}$
$\text{Houston}_{cell_{17}}$	$568_{cell_{18}}$	$261_{cell_{19}}$	$-54_{cell_{20}}$
$\text{Philadelphia}_{cell_{21}}$	$503_{cell_{22}}$	$431_{cell_{23}}$	$-14_{cell_{24}}$

The structural table, T^{struc}, is as follows.

$T^{struc}=$ {

$\langle cell_0, cell_3, \emptyset\rangle$, $\langle cell_0, cell_4, \emptyset\rangle$,
$\langle cell_2, cell_5, \emptyset\rangle$, $\langle cell_2, cell_9, \emptyset\rangle$,
$\langle cell_2, cell_{13}, \emptyset\rangle$, $\langle cell_2, cell_{17}, \emptyset\rangle$,
$\langle cell_2, cell_{21}, \emptyset\rangle$, $\langle cell_3, cell_6, \emptyset\rangle$,
$\langle cell_3, cell_{10}, \emptyset\rangle$, $\langle cell_3, cell_{14}, \emptyset\rangle$,
$\langle cell_3, cell_{18}, \emptyset\rangle$, $\langle cell_3, cell_{22}, \emptyset\rangle$,
$\langle cell_4, cell_7, \emptyset\rangle$, $\langle cell_4, cell_{11}, \emptyset\rangle$,
$\langle cell_4, cell_{15}, \emptyset\rangle$, $\langle cell_4, cell_{19}, \emptyset\rangle$,
$\langle cell_4, cell_{23}, \emptyset\rangle$, $\langle cell_1, cell_8, \emptyset\rangle$,
$\langle cell_1, cell_{12}, \emptyset\rangle$, $\langle cell_1, cell_{16}, \emptyset\rangle$,
$\langle cell_1, cell_{20}, \emptyset\rangle$, $\langle cell_1, cell_{24}, \emptyset\rangle$,
$\langle cell_5, cell_6, \emptyset\rangle$, $\langle cell_5, cell_7, \emptyset\rangle$,
$\langle cell_5, cell_8, \emptyset\rangle$, $\langle cell_9, cell_{10}, \emptyset\rangle$,
$\langle cell_9, cell_{11}, \emptyset\rangle$, $\langle cell_9, cell_{12}, \emptyset\rangle$,
$\langle cell_{13}, cell_{14}, \emptyset\rangle$, $\langle cell_{13}, cell_{15}, \emptyset\rangle$,
$\langle cell_{13}, cell_{16}, \emptyset\rangle$, $\langle cell_{17}, cell_{18}, \emptyset\rangle$,
$\langle cell_{17}, cell_{19}, \emptyset\rangle$, $\langle cell_{17}, cell_{20}, \emptyset\rangle$,
$\langle cell_{21}, cell_{22}, \emptyset\rangle$, $\langle cell_{21}, cell_{23}, \emptyset\rangle$,
$\langle cell_{21}, cell_{24}, \emptyset\rangle$

}

The following statements are true of T^{struc}.

- $cell_0 \succ cell_3$, $cell_3 \succ cell_{18}$.
- $cell_0 \succ *cell_{18}$.
- $| cell_0 |^*_{\Downarrow} = \{ cell_3, cell_6, cell_{10}, cell_{14}, cell_{18}, cell_{22}, cell_4, cell_7, cell_{11}, cell_{15}, cell_{19}, cell_{23} \}$.
- $\overrightarrow{cell_{18}} = \{ \{ cell_3, cell_0 \}, \{ cell_{17}, cell_2 \} \}$

5.5 Representation: Semantics

The semantic representation consists of:

1. Relation Semantics.
2. Inter-Cell Relationships.

Relation Semantics As mentioned earlier, the relational view of the table involves categories. The hierarchical categories used here bear some similarities to those described by Wang in [Wan96] (introduced in Section 3.1), Cameron ([Cam89]) and the 'domains' of [DHQ95].

Before arriving at the the definition of T^{RelSem} some preliminaries:

definition:
A category is a triple $\langle I, H, S\rangle$ where I is a unique identifier, H is the head ($\in \mathcal{DOM}$), possibly empty ($\oslash$) and S is a set, possibly empty ($\emptyset$), of subcategories.

The head contains a representation of the contents of the cell or cells which render the category in the table. If the category is recapitulated then there will be more than one cell which contains a string describing in some way the category head. If the category is not recapitulated then there will be a single cell and hence a single string. The role of the category can be viewed as a component of the relational view of the table and as such the strings (or whatever) in the cells have not yet been fully analysed. The analysis is completed in the final stage of the model which deals with inter-cell relationships. However, as the component of the head representing the content of the cell is not identical to the content of the cell, i.e. it is not a copy of the string as this would on the one hand be impossible if there are more than one string forms in a recapitulated category and on the other hand would not cleanly separate the semantic from the syntactic view of the table, we must define what the head actually is.

Formally, we can define a set of semantic objects $\mathcal{DOM}$ such that for each cell there is a semantic object which is represented by the contents of the cell. i.e. $\forall C \in T^{phys}, [\![C.string]\!] \in \mathcal{DOM}$. We can also say, for simplicity, that $[\![C.string]\!]$ is the same as $[\![C]\!]$. Completing this definition requires that we indicate the type

of objects which exist in $\mathcal{DOM}$. For our purposes, we might adopt a second order language capable of representing anything from individual objects to representations of incomplete linguistic analysis. The type of individual objects is something which will be further discussed when the relationships which hold between them, as well as other types of relationships between cell contents, is introduced in Section 5.5.

For convenience, we indicate the members of the set of semantic objects by a normalised form of the strings found in the cells.

1. Categories exist in either **conjunctive** sets or **disjunctive** sets. An account of the categories in a table is provided by two sets: $\mathcal{CON}$ the set of conjunctive sets, and $\mathcal{DIS}$ the set of disjunctive sets.

2. A **terminal category** is a category with a non-empty head and an empty set of subcategories.

3. C' is the **parent** of C if $C \in C'.subcat$.

definition:
For category C, the category path ($\vec{C}$) is defined as $\{C\} \cup$ the category path for C' the parent of C if one exists, and empty otherwise.

1. Categories C and D are **mutually exclusive** if the category path $\vec{C}$ and the category path $\vec{D}$ don't contain any common members.

2. A **category reading** is a set of mutually exclusive terminal categories.

definition:
The relational table, is a tuple $\langle \mathcal{CON}, \mathcal{DIS}, F \rangle$ where $\mathcal{CON}$ is a set of conjunctive category sets and $\mathcal{DIS}$ is a set of disjunctive categories, and F is a mapping from the set of category readings to the set $\mathcal{DOM}$ of interpretations of the contents of data cells. For each mapping, the tuple composed of the category reading and the data cell interpretation is called the relational reading.

1. cell $X \in T^{phys}$ is a cell member of category A ($X \Subset A$)if $[\![X]\!] = [\![A.head]\!]$.

2. $|\ \sigma\ |$, where σ is a non empty set of categories, is the set of cell members of each category in σ;

3. $[A]_{\Downarrow}$ where A is a category is the set of categories immediately below A.

Example **Table (1.4)** is again used to illustrate the representation.

(1.4)

	MURDERS$_{cell_0}$		PERCENT CHANGE$_{cell_1}$
CITY$_{cell_2}$	1990$_{cell_3}$	1996$_{cell_4}$	
New York$_{cell_5}$	2, 245$_{cell_6}$	984$_{cell_7}$	-56%$_{cell_8}$
Los Angeles$_{cell_9}$	983$_{cell_{10}}$	688$_{cell_{11}}$	-30$_{cell_{12}}$
Chicago$_{cell_{13}}$	854$_{cell_{14}}$	791$_{cell_{15}}$	-7 $_{cell_{16}}$
Houston$_{cell_{17}}$	568$_{cell_{18}}$	261$_{cell_{19}}$	-54$_{cell_{20}}$
Philadelphia$_{cell_{21}}$	503$_{cell_{22}}$	431$_{cell_{23}}$	-14$_{cell_{24}}$

$\mathcal{CON}$ ={

{

⟨cat_0, CITY, {

⟨cat_1, New York, ∅ ⟩,

⟨cat_2, Los Angeles, ∅ ⟩,

⟨cat_3, Chicago, ∅ ⟩,

⟨cat_4, Houston, ∅ ⟩,

⟨cat_5, Philadelphia, ∅ ⟩

}

⟩

}

}

$\mathcal{DIS}$ ={

⟨cat_6, MURDERS, {⟨cat_7, 1990, ∅ ⟩,

⟨cat_8, 1996, ∅ ⟩ }

⟩,

⟨cat_9, PERCENT CHANGE, ∅ ⟩

}

The mapping to $\mathcal{DOM}$ is as follows.

{

$\overrightarrow{cat_1}, \overrightarrow{cat_7} \Rightarrow [\![cell_6]\!]$ $\overrightarrow{cat_2}, \overrightarrow{cat_7} \Rightarrow [\![cell_{10}]\!]$

$\overrightarrow{cat_3}, \overrightarrow{cat_7} \Rightarrow [\![cell_{14}]\!]$ $\overrightarrow{cat_4}, \overrightarrow{cat_7} \Rightarrow [\![cell_{18}]\!]$

$\overrightarrow{cat_5}, \overrightarrow{cat_7} \Rightarrow [\![cell_{22}]\!]$ $\overrightarrow{cat_1}, \overrightarrow{cat_8} \Rightarrow [\![cell_7]\!]$

$\overrightarrow{cat_2}, \overrightarrow{cat_8} \Rightarrow [\![cell_{11}]\!]$ $\overrightarrow{cat_3}, \overrightarrow{cat_8} \Rightarrow [\![cell_{15}]\!]$

$\overrightarrow{cat_4}, \overrightarrow{cat_8} \Rightarrow [\![cell_{19}]\!]$ $\overrightarrow{cat_5}, \overrightarrow{cat_8} \Rightarrow [\![cell_{23}]\!]$

$\overrightarrow{cat_1}, \overrightarrow{cat_9} \Rightarrow [\![cell_8]\!]$ $\overrightarrow{cat_2}, \overrightarrow{cat_9} \Rightarrow [\![cell_{12}]\!]$

$\overrightarrow{cat_3}, \overrightarrow{cat_9} \Rightarrow [\![cell_{16}]\!]$ $\overrightarrow{cat_4}, \overrightarrow{cat_9} \Rightarrow [\![cell_{20}]\!]$

$\overrightarrow{cat_5}, \overrightarrow{cat_9} \Rightarrow [\![cell_{24}]\!]$

}

The following statements are true of T^{RelSem}.

- $cell_2$ is a cell member of cat_0.
- $| \{cat_7, cat_8\} | = \{ cell_3, cell_4 \}$.
- $[cat_6]_{\Downarrow} = \{ cat_7, cat_8 \}$.

Inter-Cell Relationships In the description of categories we defined a set $\mathcal{DOM}$ of semantic objects. As the table often *demonstrates* certain relationships between the components it displays (for example a 'type of' relationship holding between a category and its sub-categories), we complete the model of the table by providing an encoding of these relationships between semantic objects — the inter-cell relationship (inter-cell relationship). In the simple case, the semantic objects may be individual objects and consequently the relationship is often 'type of'. In more complex cases, the relationship may be more complex, e.g. something like a temporal adverbial modifying a category of data with the time at which it was collected. Finally there may be relationships which are expressed in the document and may in some sense be completely arbitrary.

To allow for underspecification, and to permit robust algorithmic processing in the instantiation of relationships, the inter-cell relationship is a typed object with sub-types. Inter-cell relationships are types represented by a single tree.

definition:
An inter-cell relationship type is a tuple $\langle N, S \rangle$ where N is the name of the relationship and S is the set, possibly empty, of sub-relationships.

definition:
Inter-Cell Relationships, $\langle R, X, Y \rangle$. are directed binary relationships between the members of the set of semantic objects $\mathcal{DOM}$. The simple representation is a statement of R, the type of relationship and the semantic objects X and Y. The inter-cell relationship model of the table, T^{ICR}, is the set of such relationships.

Table **Table (1.4)** is used to illustrate the above definition.

(1.4)

	MURDERS$_{cell_0}$		PERCENT CHANGE$_{cell_1}$
CITY$_{cell_2}$	1990$_{cell_3}$	1996$_{cell_4}$	
New York$_{cell_5}$	2, 245$_{cell_6}$	984$_{cell_7}$	-56%$_{cell_8}$
Los Angeles$_{cell_9}$	983$_{cell_{10}}$	688$_{cell_{11}}$	-30$_{cell_{12}}$
Chicago$_{cell_{13}}$	854$_{cell_{14}}$	791$_{cell_{15}}$	-7 $_{cell_{16}}$
Houston$_{cell_{17}}$	568$_{cell_{18}}$	261$_{cell_{19}}$	-54$_{cell_{20}}$
Philadelphia$_{cell_{21}}$	503$_{cell_{22}}$	431$_{cell_{23}}$	-14$_{cell_{24}}$

This table can be described by the following relationships (assuming the inter-cell relationship types NOMINAL_SUPER-TYPE and TEMPORAL).

T^{ICR}={
⟨NOMINAL_SUPER-TYPE, ⟦CITY⟧, ⟦New York⟧⟩,
⟨NOMINAL_SUPER-TYPE, ⟦CITY⟧, ⟦Los Angeles⟧⟩,
⟨NOMINAL_SUPER-TYPE, ⟦CITY⟧, ⟦Chicago⟧⟩,
⟨NOMINAL_SUPER-TYPE, ⟦CITY⟧, ⟦Houston⟧⟩,
⟨NOMINAL_SUPER-TYPE, ⟦CITY⟧, ⟦Philadelphia⟧⟩,
⟨TEMPORAL, ⟦MURDERS⟧, ⟦1990⟧⟩,

⟨TEMPORAL, ⟦MURDERS⟧, ⟦1996⟧⟩,
}

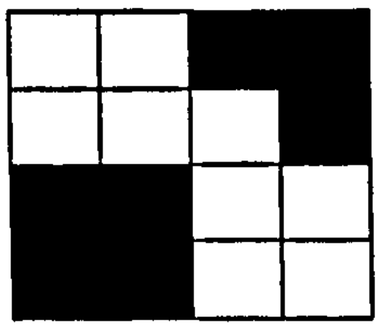
5.6 Representation: Extended Definitions

The above definitions, and the additional terms and expressions defined, are dependent only on the elements of the model under consideration. There are a few additional definitions to be made which use a mixture of the model elements.

- a **reading** of a table is the set of readings for all cells contained in $\mathcal{D}$.
- a set of cells $C \subseteq T$ is **functionally contiguous** if $\forall X \in C, \exists Y \in C : X \bowtie Y \wedge X, Y \in \mathcal{D} \in T^{func} \vee X, Y \in \mathcal{A} \in T^{func}$.
- a set of cells $C \subseteq T$ is **maximally functionally contiguous** if C is functionally contiguous and $\neg\exists X \in Tab, \exists Y \in C : X \bowtie Y \wedge X, Y \in \mathcal{D} \in T^{func} \vee X, Y \in \mathcal{A} \in T^{func}$.

  The following table contains 2 maximally functionally contiguous areas with a data cell classification.

(5.1)



5.7 Exploiting the Model

The utility of the representation is twofold. Firstly, we can use it to constrain the model space through providing an account of the interaction between model elements, thereby completing our definition of the table. Secondly, it can be used to express some heuristics which help to identify instantiations of model elements based on the existence of other elements.

5.7.1 Constraining the Model

- A data cell can only be a sink.

$$\neg\exists X \in \mathcal{D} : \exists Y \in T \wedge \langle X, Y, R\rangle \in T^{struc}$$

- A data cell must be a sink.

$$\forall X \in \mathcal{D} : \exists Y \in T \wedge \langle Y, X, R\rangle \in T^{struc}$$

- An access cell must be a source for at least one cell.

$$\forall X \in \mathcal{A} : \exists Y \in T \wedge \langle X, Y, R\rangle \in T^{struc}$$

- Cells in the STR must be aligned either horizontally or vertically.

$$\forall X, Y \in T : X \succ Y \vdash X \Leftrightarrow Y \vee X \Updownarrow Y$$

- Siblings have equal relationships with their parent. $\langle A, X, R_0\rangle \in T^{struc} \wedge \langle A, Y, R_1\rangle \in T^{struc} \wedge \exists A' \in [X] : A \Subset A' \Rightarrow A' \in [Y]$. If A is linked to X in the structural table, and A is linked to Y in the structural table (i.e. if they are both dominated by A) and the A is a cell member of A' a category in the category path of X then A' must also be in the category path for Y. This is the **distribution rule.**

5.7.2 Model based Heuristics

The use of the model to define and implement heuristics to determine new facts about a particular table is discussed in the section on the table processing system. Essentially, the model allows us to formulate in general terms any heuristic rules which might be required for a system. For example, we can make assertions based on observations in the following manner.

- A cell in the left margin is an access cell.

$$\forall X \in Tab : \text{left margin}(X) \rightarrow X \in \mathcal{A}$$

- A cell in the top margin is an access cell.

$$\forall X \in Tab : \text{top margin}(X) \rightarrow X \in \mathcal{A}$$

In addition, information may be propagated by exploiting distributional rules. For example, if a relationship is determined for a subset of the siblings of a cell, then it is implied for all the siblings as per the inter-cell relationship distribution rule. From an implementational standpoint, this means that it is possible to discover new relationships between cell elements based on information from a resource (such as WordNet [Hea98]) which fails to cover the entire content of the table. Of course, this also has implications for updating and improving such resources.

5.8 Organisation and Restriction, Rendering Structure in Tables

The model is applied to the phenomena presented in Appendix A and the analyses are presented in Section 3 of that Appendix.

5.9 Delivering Information for Interpretation

In this section we briefly look at what an instance of the model so far described, as produced by a table processing system such as that implemented and discussed in Part III, provides for subseqent processes.

As we require a table processing system to provide an analysis up to a description of the relational semantics of the table (page 137), the main output of such a system is that illustrated by the worked example on page 140. This structure describes, for each data cell in the table, the intersection of categories which describe it.

Any subsequent process may then refer to the description of categories as supplied by the relational semantics analysis to get at the textual content used to describe the path through the category structure (page 139). Consequently, subsequent processes will have access to the the set of data cells in the table and the set of strings (i.e. the textual description of each category) used to describe those data values as well as an indication of the interaction between the different categories.

These essentially form the 'table sentences' which may be extracted from the table and which must then undergo further linguistic analysis to provide a full logical interpretation.

Beyond the requirements of the table processing system as described in this thesis, the relationships between category and sub-category as described for example on

page 141, may be exploited when an analysis of the textual description of categories is carried out.

5.10 Chapter Summary

This chapter was concerned with advancing a model of tables suitable for the information extraction task. A successful model will facilitate the automatic construction of semantic information which may then be exploited within the framework of a general information extraction system.

Firstly, an indication of what might constitute a model of tables was presented. It was argued that an integrated ontology is required to describe particular aspects of the table which combine to deliver the desired semantic description. In addition, a suitable rigorous representation was also presented in the list of desiderata.

The model which was presented contained physical, structural, functional and semantic components. In addition to these basic ontological elements, further representation devices were introduced which are created through combinations of the four elements and which facilitate the interpretation of the table: the reading of the table is calculated from the functional description and the simple table relation; maximal dependency sets require inspection of semantic information for disambiguation before the categories of the relational semantic description may be determined.

Summary of Part II

Part II has collected a catalogue of the phenomena discovered in tables and presented these phenomena in terms of a layered model of the table. All of the phenomena have been discovered in a corpus of tables.

Part III

TabPro: A Table Processing System

A system is designed which processes tables for information extraction. Additionally, the design and collection of a corpus of table-including documents is presented. The system is evaluated over the corpus.

Chapter 6

Designing and Collecting a Corpus of Table Documents

This chapter presents a discussion of issues related to a general markup strategy for tables in text. The ultimate aim of this chapter is to provide an SGML DTD *capable of encoding tables in accordance with the model as presented above. It also aims to provide a summary markup scheme for documents containing tables to allow them to be processed by the system presented later in this thesis. A secondary aim is to demonstrate the scope of the problems that table markup presents (particularly with respect to the familiar in-line markup of documents already common to the* SGML *community). In addition to a description of the markup issues, a discussion of the actual task of getting documents, from either electronic or paper sources, set up in a suitable manner and marked up ready for use is presented.*

6.1 Markup For Development and Run-time Processing

The development of any reasonably complex document processing system may witness several conceptual and architectural revisions. Consequently, a vital component of the development environment is a corpus resource which is capable of serving two purposes:

1. Model input: the system requires input in order to test the soundness of implementation.

2. Evaluation: a measure of how well the system is performing is required to monitor progress and the success or otherwise of particular modifications.

Consequently, the markup for a system has two aspects. The first is that which provides a canonical form of input. The second is that which provides a description of (model) solutions to particular instances of the problem in order to facilitate automatic evaluation. We want to be able to see how well changes to the system affect performance.

6.2 Standard Generalised Markup Language

Standard Generalised Markup Language (SGML) is a textual system for marking up documents. The document type definition (DTD) provides a definition of a class of documents and is essentially a context free grammar based on a system of generic identifiers ([Gol90], p. 8). The tags which these generic identifiers denote can be augmented with a system of attributes which specialise tags and allow for referencing.

In general an SGML DTD is used to define a partitive hierarchy of 'logical'[1] elements for a particular type of document. This hierarchy might, for example, define a book as consisting of a number of chapters, each chapter consisting of a title and a number of sections, each section consisting of a header and a number of paragraphs and so on.

The depth to which the markup extends into the document varies according to the nature of the document and the application domain. If the application is to store and retrieve documents according to the presence of words in the abstract then a very brief markup would be appropriate, perhaps concentrating more information on the abstract and less on the body of the text. If the document were in fact the output of a complicated analysis then it might contain very detailed information about each word or character (for example the lexical root, stress patterns in a corpus of transcribed speech and so on).

There are two general modes of using markup. The first is to provide an *in-line description* of the type and location in a hierarchy of the elements of a text where the elements are defined to be some irreducible span of text (*e.g.* sentence, paragraph).

[1] We conform with the field by using the term 'logical structure'. However, it is not clear that this is the correct label for the what might be better called the abstract table or the abstract structure of the table.

The second mode is to provide *additional information* which is not 'present' in the raw document. This may be the output of some form of analysis as mentioned earlier.

6.3 Table Markup Systems

Currently, there are no proposed markup systems which are capable of describing anything more than the layout features of a table (though [Tho93a] provides a classification of a number of types of cell and characterises aspects of their organisation, it doesn't go far enough in its definition of the abstract/logical table)[2]. The table facilities provided by complex document markup systems like that of the Text Encoding Initiative ([TEI95]) and the CALS table markup system ([Div00]) are only capable of providing a description of the physical nature of the table, with less or more presentational information encoded.

A simple system such as that suggested by Cameron demonstrates the type of markup system which these larger systems are capable of. Cameron ([Cam89], p. 32) suggests the following outline of an SGML based DTD for tables[3]:

<!ELEMENT	table	(*table head*, table body, *table foot*)>
<!ELEMENT	table body	((cell)*)>
<!ELEMENT	cell	(top, bottom, left, right, contents, *border style*>
<!ELEMENT	top	(*integer*)>
<!ELEMENT	bottom	(*integer*)>
<!ELEMENT	left	(*integer*)>
<!ELEMENT	right	(*integer*)>
<!ELEMENT	contents	(*figure*|*equation*|text|table)>
<!ELEMENT	text	(attrib, CDATA)>
<!ELEMENT	attrib	(*font style*, *alignment*)>

The purpose of an SGML markup system for a particular document class is to:

> ... show the structural relationships among the elements of the document. ([Gol90], p. 26)

[2]Appendix C indtroduces and summarises a number of existing markup systems

[3]The DTD fragments in this document are not always complete as, for brevity, the minimisation tokens are omitted.

The structural (partitive) relationships which hold between the elements of a linear text can easily be represented by a hierarchy; however this is not the case for the same type of relationships which exist in tables.

6.4 Tables, Hierarchies and In-Line Markup

Work in document understanding usually introduces two different forms of structure:

1. Physical Structure.

2. Logical Structure.

We can illustrate the difference by considering the case of sections and subsections. Physically, a subsection of a section is merely a similar area of the document which follows from the section; logically there is a superordinate-subordinate relationship which exists between the two. This relationship tells the reader something about the text: it indicates the context of its interpretation and demonstrates a relationship with all the other elements above, below and on a par with it.

It is clear that a description of a table which uses a co-ordinate system, or row or column orientated markup ([Cam89], [TEI95], [Div00]) represents the physical layout of the table. They don't contain any information about the organisation of the cells with respect to each other. Even the row and column based strategies, though accidentally reflecting certain aspects of the underlying logical structure of the table, don't actually encode any rigorous model. As discussed in detail in Part II, though the physical alignment of cells in a table may be indicative of certain groupings and hierarchies of cells, they are not unique and unambiguous denotations.

In fact, the presence of tables in text, and the way in which they are marked up, demonstrate the lack of definition of the task which SGML markup systems have been designed to solve. They force the in-line markup strategy to change its semantics from logical (hierarchical) structure markup to layout (physical) markup.[4]. The fact that this has passed for the most part unnoticed into the large attempts to standardise the use of SGML is a little worrying.

[4]Here, we use the term 'in-line' to describe a markup strategy which maps the irreducible parts of the document logically described to those in the physical document. Clearly, in the case of tables which contain cells where the text wraps, markup which has as its basic unit the contents of a cell is not strictly 'in-line'.

The discussion about the 'logical structure of tables' is one which is generally avoided by researchers in the document understanding field as there are some very ingrained assumptions about the interpretation of this phrase, the possible complexity of tables (usually only very simple tables are considered) and so on. This chapter will not go into this issue, but in order to support the claim that the logical structure of tables cannot be marked up in-line, the work of Wang and Wood (e.g. [Wan96]) is mentioned. This work presents the most complete model of the abstract structure of tables to date as well as describing the non-physical structure of tables as being the *abstract table*. The term 'logical' is an unfortunate misnomer which is hard to avoid.

Leaving the details of the model which Wang presents aside, the point which we wish to make is that a real description of the abstract table requires that certain elements in the table be mentioned more than once in order to define certain relationships between cells and other higher order tabular structures (hierarchical organisations of cells). Consequently, some form of reference must be employed if the *contents* of the table are to be marked in-line and the 'logical structure' is to be represented.

6.5 System Requirements

There are two general issues that must be addressed when considering the design of the markup. The first is the obvious: what is to be marked and how? The second is related to interest in the robustness of the application. Document understanding systems, text retrieval systems and information extraction systems when fortunate can deal with clean text created in an electronic format. However, it is often the case the documents must be scanned in, the document image processed by optical character recognition software and the resulting text stored. A fully operating system will not have the original clean document to work with and must be robust over the noisy version.

Though it is not necessarily in the scope of this thesis, the noisy scanned version of the document must be accessible as either simple text or preferably as a fully marked up document. Additionally, it must be possible to log any corrections made to the noisy document. This leaves two possibilities. The first is an interweaving of the original document and the clean document. The clean document would exist

as a series of corrections to the original document and could be recovered by some sort of filter operation. The second involves a certain amount of redundancy and is a pair of documents, one being the clean marked up document and the second being the original document marked up.

In the interest of perspicuity, it would seem that the second of these two choices would be more useful. It has the advantage of being (more) readable by humans and also involving less work on the part of the computational system. This decision is also influenced by the number of changes which must be made to a scanned document before it conforms with the content of the original. Tests have indicated that there are many errors in the scanned in documents and so there would be no advantage to resources in interleaving the original and corrected documents. Additionally, the source of the document may not be a scanned in document or simply may not be available.

A complete account of a single document, then, might involve a description of the scanned in document, marked up with alterations and the markup required for the application and a description of the complete document. As mentioned above, due to the mode of the document, there may not exist a scanned version, additionally, producing both a marked up original and a clean version represents either a large effort on the part of the analyst or a complex document handling system, or both. Consequently, the corpus version of a document should consist of a clean marked up document, an optional purely textual version of the original scanned version and an optional marked up version of the scanned version.

6.5.1 Header Information

Information describing the document is stored in a header. This should contain firstly information about the source document (e.g. the paper document).

- The title.
- The author.
- The organisation.
- The address.
- The date.

- The source type, either paper or electronic.

```
<!ELEMENT     title          (#PCDATA)>
<!ELEMENT     author         (#PCDATA)>
<!ELEMENT     org            (#PCDATA)>
<!ELEMENT     add            (#PCDATA)>
<!ELEMENT     date           (#PCDATA)>
<!ELEMENT     source         (#PCDATA)>
```

Secondly, information about how the document was obtained and how it has resulted in the version included in the corpus.

- How obtained.
- Obtainer.
- Date obtained.
- Marker.
- Date of initial version.

```
<!ELEMENT     obtain         (#PCDATA)>
<!ELEMENT     obtainer       (#PCDATA)>
<!ELEMENT     date           (#PCDATA)>
<!ELEMENT     marker         (#PCDATA)>
<!ELEMENT     vers0          (#PCDATA)>
```

Thirdly, a revision log of alterations made since the date of the initial version. The revision log contains information about when the alterations were made, by whom, and the location of the alterations in character positions in the file.

- Date of alteration.
- Altered by whom?
- Alterations:

– Start of alteration in previous document.

– End of alteration in previous document.

```
<!ELEMENT    altlog          (date, person, altlist)+>
<!ELEMENT    altlist         alter+>
<!ELEMENT    alter           EMPTY>

<!ATTLIST    alter    start  NUMBER    #IMPLIED
                      end    NUMBER    #IMPLIED>
```

6.5.2 General Document Markup

As the documents we are interested in contain tables mixed with text, we must provide some general markup for the non table parts of the document. We will take the paragraph as the basic element (which may be interspersed with references to the tables). Above this we will recognise the hierarchical division of the document into sections and so on.

```
<!ELEMENT    body            section+>
<!ELEMENT    sheading        #PCDATA>
<!ELEMENT    section         (sheading, text*, subsection*)>
<!ELEMENT    subsection      (sheading, text*, subsubsection*)>
<!ELEMENT    subsubsection   (sheading, text*)>
<!ELEMENT    text            (p|table)>
<!ELEMENT    p               (#PCDATA|(#PCDATA?, tref, #PCDATA?))+>
```

A more elegant solution, which would avoid the depth restriction of the generic indicators `section`, `subsection` and so on would be to define a general purpose `section` tag and have an attribute describing the depth of the hierarchy. However, there is no way to ensure that the depths are incremental and ordered. For example, in scanned in documents, if the document is to be automatically translated into the markup described here the sections must be located and their headings marked as such. If the DTD represented a truly logical hierarchical section-subsection structure

then the numbering system would need to be identified and encoded in a consistent manner.

6.5.3 Table Reference

Tables can appear in documents in two different ways: in-line or floating. In-line tables don't have labels (though they may have captions) whereas floating figures require labels so that they can be identified. References to floating figures must have labels in them which match the label of the appropriate figure (label reference) whereas references to in-line tables contain phrases which locate the table in the stream of the document or the argument structure of the text (discourse reference).

The markup of references to tables should in some way indicate the table in the marked up document to which they refer.

Label Reference

Label references generally consist of a phrase mentioning the table and the reference in whatever numbering strategy is appropriate to the document class of the form

table N

As, in terms of the text, this reference is a noun phrase, it is subject to the grammatical possibilities that any noun phrase is. For example, conjunction:

- table N and table M
- tables N and M

consequently, the most appropriate point of reference is simply the text which indicates the table by the numbering strategy of the document type. Otherwise problems would occur in the above situation.

Discourse Reference

There are two types of reference which fall into this category. The first is those phrases which use the word `table`. The second is those which don't.

- `the following table.`

- `the following.`

An analysis of the corpus provided a summary of the ngram contexts surrounding references to tables and appears in Figure 3.3 in Chapter 3.

Reference Markup

Clearly, both types of reference could be subject to a large amount of linguistic analysis which would help in designing a comprehensive markup system. As there is neither the time nor space for this now we must design a means of marking up reference which can be implemented now and which may be incorporated into a larger and more complex strategy. Consequently, the references to tables should be marked as either the text of the numbering strategy for label references and the smallest appropriate noun phrase for the discourse references.

```
<!ELEMENT     tref                    (#PCDATA)>

<!ATTLIST     tref          refid     IDREF          #IMPLIED
              trtype        (label|discourse)        label>
```

6.5.4 Table Markup

As discussed above, a table's logical structure cannot be represented by the traditional in-line markup methods usually associated with SGML document markup. Clearly, though, the contents of the table have to be marked up in some way. The strategy adopted here is to include in the description of the table a description of the layout. The 'cells' in the table can then be referenced by subsequent complex abstract descriptions of the organisation of the tabular material. In this way, the simple markup strategies of other systems may be followed and combined with more useful semantic information. The following markup, then, is an encoding of the physical table (Section 5.2).

```
<!ELEMENT table              - -      ((tabledata & title &
                                      tablehead? & tablefoot?), analysis?)>
<!ATTLIST table        id    ID       #REQUIRED>
```

```
<!ELEMENT titletitle        - -     (tablelable? & caption?)>

<!ELEMENT tablelable        - -     (#PCDATA)>

<!ELEMENT caption           - -     (#PCDATA)>

<!ELEMENT tabledata         - -     (cell+)>

<!ELEMENT cell              - -     (cellcontents, celldescription)>
<!ATTLIST cell      ID      ID      #REQUIRED
                    X1      CDATA   #REQUIRED
                    Y1      CDATA   #REQUIRED
                    X2      CDATA   #REQUIRED
                    Y2      CDATA   #REQUIRED>
```

The above schema represents a table as a series of cells. These cells are positioned with a relative co-ordinate system. The x-axis is horizontal from left to right and the y-axis is vertical from top to bottom. The numbers represent the relative co-ordinates of the top-left and bottom-right corners of the cell. The co-ordinate values start at 1 and are based on minimal divisions of the table in the x and y dimensions. For example, if a table consisted of a single cell, the co-ordinates would be {(1,1), (1,1)}. If the table consisted of two columns with a single spanning header, the co-ordinates of the header would be {(1,1), (2,1)}. In this case, the x co-ordinate is incremented to accommodate the extra division in the columns. The co-ordinate of the first cell in the second (right-hand) column would be {(2,2), (2,2)}.

Now that the layout of the table has been marked, we can use the references to the cells (the SGML ID tags) as part of the description of the abstract structure. Additionally, as this markup system is to be used for evaluating the development of the system, we can code up various intermediate results which the system can use to check its progress at various stages. The simple table relation is as follows

```
<!ELEMENT     str           - -     (cellrec+)>
<!ELEMENT     cellrec       - -     (cellpair, restrictionlist,
```

```
                                                        note)>

<!ELEMENT     cellpair                  - -         EMPTY >
<!ATTLIST     cellpair      idone       IDREF       #REQUIRED
              idtwo         IDREF       #REQUIRED>

<!ELEMENT     restrictionlist           - -         (restriction*)>
<!ELEMENT     restriction               - -         EMPTY>
<!ATTLIST     restriction   refid       IDREF       #REQUIRED>
<!ELEMENT     note                      - -         (#PCDATA)>
```

the category description is,

```
<!ELEMENT     category                  - -         (head, subcategories)>
<!ATTLIST     category      id          ID          #REQUIRED>
<!ELEMENT     head                      - -         (#PCDATA)>
<!ATTLIST     head          cells       CDATA       #REQUIRED>
<!ELEMENT     subcategories             - -         (category*)>
```

6.6 Gathering A Corpus

The corpus was gathered from a number of sources.

1. Hand collected news articles copy typed from paper source. The first few documents that were collected during the initial stages of research were short news pieces collected from weekly news magazines. These articles consisty of a few hundred words and only one table.

2. Scanned in documents converted to text files using OCR software. These documents were collected from the BICC domain, being one of the motivations for table processing research. The application was to translate tables of descriptions of constraints for constructing multi-fibre cables into rules for an expert system. These legacy documents were in general of intermediate (image) quality and some amount of cleaning up was required.

3. LaTeX files. These files were taken from a number of sources. All the document used in that project are from the Computational Linguistics e-print archive and represent the largest collection of documents in the table corpus from single domain.

4. HTML files. Perhaps the most readily available and convenient source of documents is the web. Documents containing tables can be found using key word searches based on sentence fragments which refer to tables, or simply the word `table` itself.

The LaTeX files and the HTML files were converted to the system DTD using a purpose written perl script. The other documents were marked up by hand.

The corpus covered a wide range of domains which were catalogued as shown in Figure 6.1.

Domain	Number of Documents	Number of Tables
Administration	9	14
Biochemistry	2	2
Computational Linguistics	6	14
Construction Industry	1	13
Finance	1	3
Information Technology	8	41
Internet	1	1
News	2	2
General Science	18	72
Sport	1	1
total	49	163

Figure 6.1: Content Domains

6.7 Markup Strategies

Given that it is impractical to mark up tables, and documents, by hand using a simple text editor, a number of issues have to be dealt with when considering the

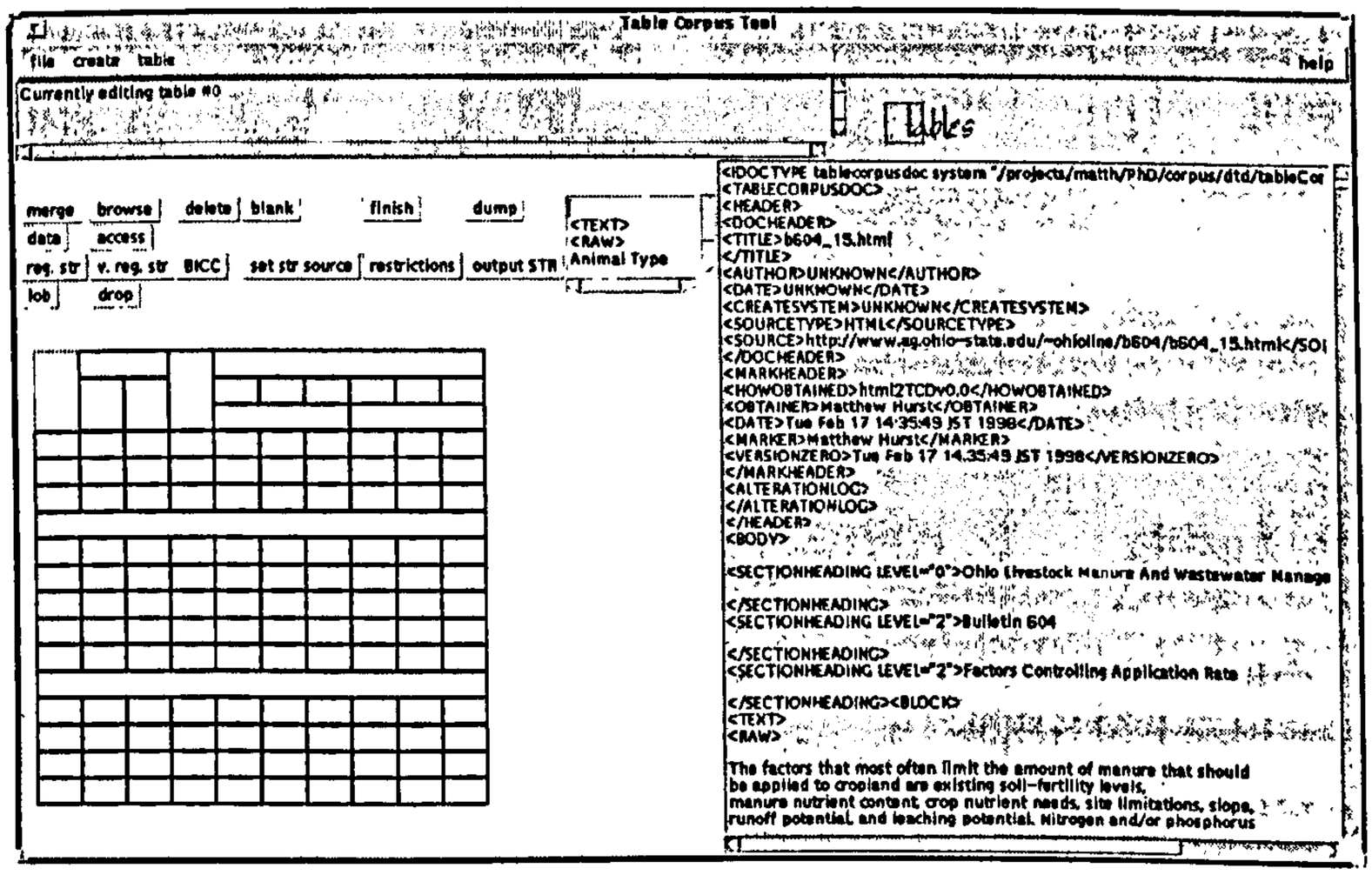


Figure 6.2: The Table Corpus Tool.

construction of the corpus. If the documents are already marked up to some degree (e.g. LaTeX or HTML) then markup can start by providing a translation from the source markup to the DTD outlined above. In the case of both LaTeX and HTML it is feasible to write a simple program in a text processing script language such as PERL to do this job.

If the source documents are raw text (ASCII), then there are two basic sub-tasks. The first is to locate the parts of text which are to be marked up in a particular manner (headers, paragraphs, tables), a task generally referred to as text zoning. The second is to mark them up appropriately. This is straightforward enough for headers and paragraphs, however for tables there is still a lot of work to be done. A number of research projects exist (e.g. [RKG95]) which deal with just this task.

Once the basic physical nature of the document has been recorded, the markup of the correct model has to take place. This provides both training data and evaluation data for table processing systems.

A tool was written in Java (Figure 6.2) which presents the corpus annotator with a visual rendering of the physical table and uses the point and click paradigm to select sets of cells and establish relationships between them. The left-hand panel presents a simple graphical description of the table in terms of cell boundaries. The

right-hand side displays the original document. Markup is carried out by selecting a cell and assigning a feature, or selecting a pair of cells and assigning a relationhips. For example, functional information is entered by selecting a cell and pressing either the data or access button.

6.8 Chapter Summary

This chapter has been concerned with two things:

- Discussing the use of SGML for document markup and how the markup of tables requires, in some respects, a re-evaluation of *what* is being marked up.
- Presenting, in fragments, the components of a DTD which provides a general markup for the corpus as well as a solution to the problem of marking up the abstract structure of a non-linear textual object.

Chapter 7

Designing and Implementing Algorithms and Resources for a Table Processing Workbench

This chapter describes the design and implementation of a system for building instances of the table model described in Part II of this thesis.

7.1 Objectives

The purpose of the Table Processing System (Tabpro) is, given a document containing one or more tables, to provide an instantiation of the table model for each table. The instantiation can then be employed by an information extraction system to retrieve the facts deployed by the table (see Section 1.4). Ideally, a system should do the following:

1. Identify examples of the class of tables it is able to process. This in itself is a non trivial problem.

2. Respond in a reasonable time. Generally, a 'reasonable time' is a factor of the system in which the table processor is embedded. However, given the speed of current information extraction systems (e.g. FASTUS, [HAB+]) we desire a response within a few seconds.

3. Provide a measure of its confidence in its results. It is important to consider the reliability of the output. Achieving this involves not simply invoking probability based performance measures of subsystems based on, for example, simple bayesian strategies, but employing some measure of our confidence in the analysis based on measures capable of characterising judgements about the model in general. For example, monitoring the ratio of access cells to data cells.

4. Be robust to input errors and poorly coded tables. Most manifestations of tables in documents are marked up or coded purely in terms of presentation (e.g. LaTeX([Lam85]), HTML ([W3C98])) or appear as simple ascii arrangements. The latter may be transformed to, for example, HTML by systems such as [KD98], though the result often requires more work to reach an ideal standard. Consequently, such factors as empty cells used for space filling, extra border spaces, single cells for multiple row or column spanning cells and so on are common. A certain standard of input should be aimed for to provide a level starting point for processing.

5. Degrade gracefully. Where analyses fail we would like the system to be able to deliver some portion of the final result with an associated confidence rather than simply deliver results blindly with no idea of their quality or coverage. Additionally, the system should be able to monitor performance and either attempt alternative strategies of processing, or notify the containing system in an appropriate manner.

As this is a research project, and there are many potential applications for intelligent table manipulating technology, we also require that the following:

- The system have a clean development API.
- The system is amenable to varying resources.
- The system is reasonably portable.

The purpose of this work is to provide technology for integrating tables with current information extraction technology. It is not the aim of this project to build large computational linguistic resources, nor to implement basic systems which are already available as mature applications from other areas of the research community.

Consequently, lexica, semantic networks, text manipulating libraries and so on will be employed as imported artifacts.

It should be noted that, in designing for experimental work, the system is more flexible than the experiments described later may suggest.

7.2 Implementation Strategy

The flexible design implies certain control structures for the system, most importantly a hypothesis manager for organising the results of various processes. Additionally, it suggests the organization of resources and modules as two distinct types of system component. Resources are data or subsystems which are generic and exploited by the modules, which are specific task oriented transducers. Resources may stay the same and be used by different modules, or modules may be constant and resource may change.

The experimental requirement also suggests a control loop based on some form of script by which the user may interact with and inspect the system state during development and use.

The remainder of this chapter will describe

- a broad overview of the system architecture (Section 7.3).
- the manner in which documents are input to the system (Section 7.4).
- the preprocessing of document before table analysis (Section 7.5).
- the list of resources (Section 7.6).
- the list of module tasks (Section 7.7).
- the management of hypotheses (Section 7.3.3, Section 7.8).
- the development of a script language for controlling and interacting with the system (Section 7.9).

7.3 System Architecture

The system contains a number of key control subsystems.

1. a resource manager.

2. a module manager.

3. a hypothesis manager.

The main focus of the system is the modules. The modules act as transducers taking one component of the table model and instantiating another. Other support processes are coded as resources which are employed by the modules and other subsystems.

7.3.1 Resource Manager

Resources are identified primarily by the service they offer. The manager offers interaction with these resources through an API. Currently the resources include:

tokeniser : The tokeniser takes as input a text string and produces a list of substrings which are deemed to be tokens according to a standard set of criteria.

lemmatiser : Given a word, the lemmatiser delivers the root.

crystaliser : The crystaliser takes as input a text string and produces as output a set of substrings which are marked as of a certain semantic type (a crystal). These types include dates, units of measure and so on.

semantic network : The semantic network is used to discover cell contents which stand in type of relationships.

table sentence extractor : The table sentence extractor discovers sentences in the document which refer to one or more tables.

7.3.2 Module Manager

Modules are identified primarily by the type of transduction they perform. The module manager is responsible for registering modules which perform a certain transduction and delivering modules when required. The manager can also check to see if the resources required by a module are available.

The following is a list of module types and instances developed for this research.

function determination : SIMFUN, PATTERNFUN, HEURISTICFUN, CONTENT-FUN.

structure determination : HEURISTICSTRUC.

relational semantics determination : SIMRELSEM.

inter-cell relationship determination : SIMICR.

7.3.3 Hypothesis Manager

A hypothesis (Appendix B.4) registered with the hypothesis manager contains an assertion (Appendix B.5) that a module wishes to make about some part of the table being processed. Hypotheses are identified by:

- the name of the asserting module;
- the function of the asserting module (the task);
- the type of the assertion in terms of the type of module posting it (Section 7.3.2);
- the time of the assertion;
- the cell(s) involved in the assertion.

Any module may ask the hypothesis manager for a subset of the hypotheses registered. The subset may be identified by any number of the following features:

- the name of the asserting module;
- the function of the asserting module;
- the type of the assertion;
- the cell(s) involved in the assertion.

Although this architectural component provides a certain amount of functionality similar to the blackboard paradigm it is less general, and the processes associated with it follow a more linear progression than that of blackboard systems.

7.4 System Input

As described in Chapter 6, documents are marked up in SGML according to the project DTD. Such documents are manipulated by the LTG NSL tools ([MTT+97]). This library performs normalisation of the document markup and supports the process of loading the document.

7.5 Document Preprocessing

Once a document is loaded a certain amount of preprocessing is performed. The preprocessing carries out certain shallow linguistic analyses of the document, the results of which are used in a contextual manner for processing the tables. The preprocessing operates over all the text in the document and consists of

1. tokenisation.
2. chunking (discovering noun groups and verb groups).
3. crystal detection.
4. detection of sentences referring to tables either explicitly or implicitly.

7.6 Resources

The resources of the system are general purpose tools and knowledge bases which are exploited by the transducer modules and other system components. There are essentially, three types of object associated with the notion of a resource:

1. The request object.
2. The result object.
3. The resource object.

The resource object must accommodate the identity of the resource (what it is called and what type of resource it is) as well as specialised procedures for fielding requests. The request object must also identify what type of request it is, as well as supplying the details of the request in an appropriate form. The result object

stores the results in a manner appropriate to the type of resource. The API for the resource is shown in Appendix B.2.

7.7 Modules

The modules are the system components which carry out the main analysis of the tables. They instantiate a component of the table model based on other table model components and the context of the table in terms of the document content.

A module has a name, an optional personal name used by any assertions it makes, a task identifier describing the task that it carries out and an indication of the confidence in the output. The module should be able to initialise, clear any variables or structures in order to start a run and manipulate any files which it requires to read from or write to.

7.8 Hypotheses and Assertions

The hypothesis must store information about its origin, including the name of the module which generated it, the task that module was carrying out, the time of the session and the time at which the information was generated. In addition it holds the confidence the module had in asserting the hypothesis and the assertion being made.

The assertion contains information about its type. Although a hypothesis stores the type of the module which generated it, as a module may generate a number of assertions, representing different components of the model element being processed, the assertion must store information about its own type.

7.9 A Scripting Language to Control Table Analysis

This section describes a scripting language implemented to run various sequences of operations when analysing tables.

The basic operation that is required of the system is to run a module with a given input and produce an output. In this system, the output is a set of hypotheses describing a component of the table model as instantiated for a particular table. Consequently, the basic command of the system is the run command. In order to

monitor the state of the system, a number of measures may be taken of the state of the analysis. Two types of variables exist. The first is a simple string variable which allows the script to define a replacement for one string by another; the second allows a value to be bound to a variable. Such a variable may then be inspected. Controlling the order and the flow of operation, the script language implements loops and conditional statements.

7.9.1 Initial Processes

Before analysis is carried out, a number of initial commands must be issued.

Module and Resource Registration

The first thing a script should do is register any modules and resources which are going to be used.

Registering Documents

Any documents which are to be processed must be registered with the system. Document registration allows the system to normalise the document markup in preparation for loading and processing. Documents may be registered with either the `document` command, or the `corpus` command. The `document` command takes a single argument, the file name (and path if required). The `corpus` command takes the name (and path is required) of a file containing a list of file names. The effect of the `corpus` command is to run the `document` command on all the files listed in the corpus file.

```
document < file_name >;
corpus < corpus_file_name >;
```

Document and Table Selection

A document from the set of registered documents is selected using the `setdoc` command. This loads in the document and carries out the preprocessing.

A document may have any number of tables. The tables can be listed by using the `tablist` command. The `settable` command is used to select the table to be

worked on.

7.9.2 Running Modules

The simplest manner in which modules are invoked is by the **run** command, followed by the module name. Of course, there may be any number of arguments required or optional with each module. All modules may take the **personal name** argument. This argument, indicated by *-p* provides a unique identifier for the module on this particular run and will be associated with any hypotheses which are asserted. The **personal name** is appended to the module name to create the associated name. For example **run** PATTERNFUN *-p test* will result in the name **PatternFun:test** being associated with the hypotheses asserted by that module.

Another common argument is a specification of the hypotheses to be used as input to the module. In general, when no hypotheses are specified, or none is required, the module will load in all the hypotheses of the required type, or none at all. If hypotheses are specified then only those asserted by the specified module and of the required type will be loaded. For example,

```
run HEURISTICSTRUC -h PatternFun:test;
```

will run the module HEURISTICFUN with the hypotheses asserted by the module PATTERNFUN when run with the **personal name test**.

7.9.3 Inspecting The System

A number of numerical characterisations of the table and the system analysis may be derived in order to make conditional decisions.

1. grid: the ratio of spanning cells to non spanning cells.
2. functional ratio: a function of the number of contiguous data areas, the number of contiguous access areas, the average and total size of those areas and the number of data and access cells. Observing the tables in the corpus allows a minimum of this measure to be computed giving a threshold against which the quality of a functional analysis may be measured.

7.9.4 Conditional Statements

Conditional statements may be made which result in a boolean value based on simple numerical inequalities. As a result, certain parts of the script may be run conditionally. Conditional statements take the form

`if [` *ineq* `]` *block*

where *ineq* is an inequality of the form `gt`, `lt`, `lteq`, `gteq` or `eq` followed by a variable and an integer or floating value. For example

`if [` *lteq %val 0.9* `] { run` SIMFUN `}`

means that if the value of the variable *%val* is less than or equal to 0.9 then run the module SIMFUN.

7.9.5 Model Conversion

A given module acts as a transducer from one component of the table model to another. For example from the physical model to the functional model. Although we view the relationship between the components in a declarative manner, it is clear that some levels of the model are semantically more sophisticated than others. This can be illustrated by the requirement for more knowledge rich resources and so on. However, it is useful to consider the alternative direction of processing from higher level components to lower level ones. A prime example is the calculation of a table's functional description from its structural description. Given a simple table relationship for a table, any cell which has no arcs exiting it and only arc entering it must be a data cell.

Converting results in this direction allows the system to check its progress based on the assumptions or constraints which hold between different model components.

7.10 Resource Descriptions

Resources are used at various times by both modules and elements of the control architecture.

7.10.1 Tokenisers

A tokeniser takes as input a string, and responds with a list of the discrete tokens found, in order, in that string. In general, this implements the notion of splitting up the words in a sentence by looking for space delimited strings, but must also be sensitive to the various special cases such as the attachment of punctuation to words and the inclusion of abbreviations.

For example, the sentence `America's five largest cities are enjoying a major drop in crime.` would result in the tokens `America`, `'`, `s`, `five`, `largest`, `cities`, `are`, `enjoying`, `a`, `major`, `drop`, `in`, `crime` and `.`.

7.10.2 Chunkers

A chunker takes as input a string and produces a list of the noun groups and verb groups found in the string. The chunker used by the system is the LTG product LTCHUNK ([Gro99]). The chunker has been imported pretty much as is, and is implemented by a client/server architecture. It returns text marked up according to the presence of noun and verb groups, and the markup is processed to provide arrays of the groups which are then returned as the result of the chunking process.

For example, the sentence `America's five largest cities are enjoying a major drop in crime.` would produce the following chunks. `America's five largest cities`, `are enjoying` and `a major drop in crime`.

7.10.3 Sentence Filters

The sentence filter provides some information about the type of sentence. In this case, we are interested in finding sentences which refer to the table which is being processed.

7.10.4 Lemmatisers

A lemmatiser produces the root of a word. For example, `cities` → `city`, `enjoying` → `enjoy`.

7.10.5 Crystalisers

It is useful to find strings in text which are easily identifyable as members of a certain semantic class. For example, units of measure, dates, numbers and so on. The job of the crystaliser is to take the input string and identify such substrings.

7.10.6 Ontological Analysers

Although a domain specific ontology is not included in the general design strategy of the system, a general purpose ontological resource has been included. Due to the consistent nature of relationships between parents and children (i.e. if X is the parent of cells A and B, then A and B will stand in the same relation to X), a low recall, high precision analysis of relationships can be successfully distributed over sibling, parent pairs. For example, if the system can deduce from the general ontological resource that Houston is a CITY, then it can infer that all the siblings of Houston (New York, Chicago, etc.) are also of type CITY.

The ontological resource installed in the system uses WordNet to identify type of relationships between cell contents.

7.11 Module Descriptions

This section describes in detail the modules which have been created for the system.

7.11.1 Functional Modules

The task of determining the functional model of a table can be defined as follows:

task definition:
For each cell in the table, determine if it is a member of $\mathcal{A}$, $\mathcal{D}$ or neither.

Bayesian Table Function Classification

The Bayesian approach to classification is well documented ([Mit97]). It relies on a set of independent features each with a discrete set of values. The task is to assign a value to a variable (v) from the target set (V) given a value for each of the features ($a_0, a_1, a_2 \ldots a_n$).

$$v = \underset{v_j \in V}{\text{argmax}} P(v_j \mid a_0, a_1 \ldots a_n)$$

which, given Bayes theorem, can be rewritten as

$$v = \underset{v_j \in V}{\text{argmax}} \frac{P(a_0,a_1,a_2 \ldots a_n | v_j) P(v_j)}{P(a_0,a_1 \ldots a_n)}$$

and

$$v = \underset{v_j \in V}{\text{argmax}} P(a_0, a_1, a_2 \ldots a_n \mid v_j) P(v_j)$$

Given the independence of the attribute values, we can calculate the probability of the conjunction of the values by using the product of the probability of each value given each classification from the target set.

$$v = \underset{v_j \in V}{\text{argmax}} P(v_j) \prod_i P(a_i \mid v_j)$$

The next step is to select a set of features and their appropriate values. To do this, we must consider the objects which we have to observe, and the methods by which we can observe them.

Calculating the functional description of the table is in some ways the kick off point for table processing, so we should, at this stage, assume that no other model components have been computed. Consequently, we have only the physical description of the table and the contents of the cells and the document to work with.

The features can, then, be split up into two broad classes. Those which derive from the physical description of the table, and those which derive from the content of the cells, the content of the document and any relationships we may observe between the two.

The physical model suggests a simple set of features — those describing the number of cells adjacent to the cell in question above, below, to the left and to the right. The value of this observation can be either null, one to one, one to many or many to one.

The content based features are more complex. First of all, a number of processes are considered for distinguishing the strings found in the document and in the table cells:

- normalisation: the string is normalised to remove repeated spaces, new lines and so on;

- tokenisation: the string is split into components delimited, in general, by spaces, though care is taken with the attachment of punctuation to strings and so on;

- chunking: the string is processed to find noun groups and verb groups;

- heads: the result of chunking is processed to recover the heads of noun or verb groups.

Although normalisation and tokenisation may be thought of as content independent, the task of extracting noun and verb information from a string is not. An important consequence of this is that, given the few words found, in general, in a cell, processing cell content strings for such information is not reliable. Consequently, once this information has been processed for the body of text found in the document surrounding the tables, the results may then be compared with the cell contents. For example, if the string run is found in the document to be a noun and we later find the string, in isolation, in a cell, we can guess that it is also a noun when it appears in the table.

Content based features, then, are divided up into token, noun group, noun head, verb group and verb head sets. For each feature group, a value is computed representing the proportion of the cell content string which is found in the set of tokens from the main text, the set of noun groups from the main text, and so on. These values are then normalised to give discrete feature values.

In addition to these basic methods, a number of areas of the document are focused on and provide their own context feature information. These areas are:

- Section Headings.

- Sentences referring to tables.

- Sentences referring to the particular table being processed.

- The caption for the table.

Identifying the section headings and table captions is trivial as they are elements of the logical document model described by the DTD. However, locating sentences which refer to the table is an interesting task in itself. A simple strategy is to look for the string `table`, as well as patterns such as `table 1`, `table A` and so on.

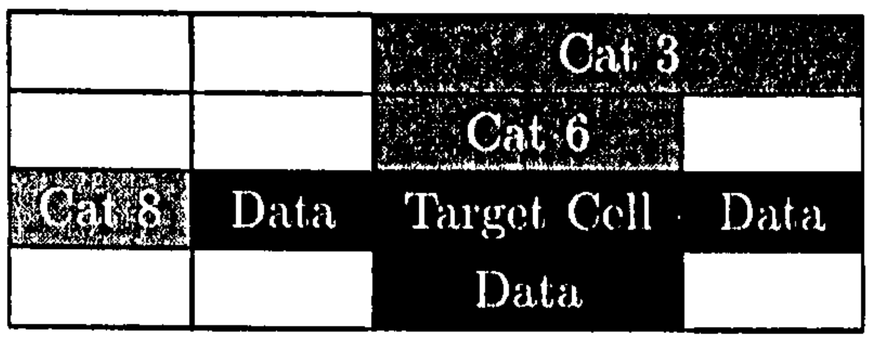
More accuracy is afforded if relative phrases are also covered such as `the following table` or `the table at the top of page 5`. Naturally, the system ought to be sensitive to documents discussing furniture and ping-pong.

Pattern Based Function Classification

Using patterns to classify the function of cells requires firstly some definition of a physical pattern in terms of the cell in question. Secondly, an algorithm to match the pattern to new instances. And finally, a classification algorithm to deal with the aggregate of patterns which may have matched in a particular instance.

The patterns used by this module are not complete patterns of the context in which a cell appears but tree like structures which are grown to a specified depth perpendicular from the four cell faces, as illustrated below where a pattern of depth 2 is described.

(!7.1)



Matching the pattern to a new instance can be achieved by a number of variations of the simple, obvious method. One important feature is whether or not matches are done in a specific or general manner (or, alternatively, if patterns are produced in a specific or general manner). For example, a pattern which includes a cell spanning 3 other cells can either be regarded, in the specific case, as spanning exactly that number of cells, or, in the general case, as simply spanning a number of cells greater than 1. For implementational reasons, it is simpler to adopt the specific matching criteria.

Additionally, we can make hard and soft restrictions regarding the behaviour of the matching algorithm at the edge of the table. We may require that the pattern be exhausted and fully matched, or that the pattern, if the table edge is encountered, is not prevented from succeeding.

Another variation might take into account the presence of cells with a function already computed by another module. We could insist that the matching algorithm be perfectly aligned with pre-existing classification information.

Once we decide on the matching algorithm, we must decide on the manner in which it performs the classification of the cells. The simplest approach is to use a voting method. For each cell, the number of patterns matching which have access to function classification are counted as are the data classifications. The majority is taken as the classification of the cell in question.

Heuristic Based Function Classification

A number of heuristics are implemented by this module indexed by the following numbers:

1. reserved index, currently not used.

2. rectilinear contiguous areas. This heuristic deals with areas of data cells in the table. If a contiguous area of data cells is found, then it should be rectilinear. If it is not, then an attempt is made to modify the classification of the cells or those which surround the area.

(!7.2)

Cat 1	Cat 2	Cat 3	Cat 4
Cat 5	Data	Data	Data
Cat 6	Data	Data	Data

is transformed to

(!7.3)

Cat 1	Cat 2	Cat 3	Cat 4
Cat 5	Data	Data	Data
Cat 6	Data	Data	Data

The algorithm, given a contiguous area of data cells, proceeds as follows:

(a) Calculate the set of cores. These are maximal rectilinear areas within the contiguous area. The example **Table (7.2)** has 2 cores, **Table (7.4)** and **Table (7.5)**.

(!7.4)

Cat 1	Cat 2	Cat 3	Cat 4
Cat 5	Data	Data	Data
Cat 6	Data	Data	Data

(!7.5)

Cat 1	Cat 2	Cat 3	Cat 4
Cat 5	Data	Data	Data
Cat 6	Data	Data	Data

(b) Calculate the bounding box. This is the box which includes all the cells in the contiguous area. The result is shown in table **Table (7.6)**.

(!7.6)

Cat 1	Cat 2	Cat 3	Cat 4
Cat 5	Data	Data	Data
Cat 6	Data	Data	Data

(c) Interpolate between the cores and the box. Each step in the interpolation is qualified according to a metric which balances the number of cells required to be changed if the interpolation where to become contiguous with the total area of resulting from the changes.

(d) Select the best interpolation according to the metric.

3. All daughters have the same type. All the cells spanned below or to the side of an access cell should be of the same functional type. **Table (7.7)** is transformed to **Table (7.8)**:

(!7.7)

Cat 1			
Data	Data	Data	Data

(!7.8)

Cat 1			
Data	Data	Data	Data

4. If all daughters are access then the parent is access too. A cell which spans a group of cells all of which are access cells should be marked as being access cells. **Table (7.9)** is transformed to **Table (7.10)**:

(!7.9)

Cat 1			
Data	Data	Data	Data

(!7.10)

Cat 1			
Data	Data	Data	Data

5. Equal strings have equal classification. If a string is found repeated in a table then all those cells containing that string should be normalised to the same functional classification.

6. Repeated strings are access cells. When a domain is recapitulated, the cells, naturally, repeat in the table. Conversely, if a particular string is observed to repeat in the table then it suggests that it is a recapitulated domain, and hence, a set of access cells.

7. Cells which span the entire table are access cells. Such cells are cut-in cells.

8. Left margin cells are access cells.

9. Top margin cells are access cells.

10. Syntacto-semantic rules. Using a set of simple Finite State rules, classes of strings can be recognised which are associated with certain simple semantic types. Rules can be written to indicate how these types are to be classified when found in particular physical arrangements.

Content Heuristic Based Function Classification

A further module using heuristics was implemented which looked only at the presence of certain types of words and phrases in specific document locations. The module checks for noun groups, noun group heads and tokens in section headings, sentences referring to tables and table captions. It then uses these strings to match the contents of table cells. If a match is found, then the cell is marked as an access cell.

Essentially, this module uses a very shallow form of discourse based processing. The simple model of table related discourse assumes that access content in the table is discussed in the sentences referring to the table and bear some relationship with the general structure of the document's discourse.

Management Level Modules

Two management level modules exist for the function classification task.

1. Vote Manager. The vote manager is a simple module which, given a list of modules which have produced hypotheses about the function of cells in the table, tallies functional assertions and asserts the most frequent for each cell.

2. Filtering. The filter module makes decision about whether or not to allow an assertion based on some pre-determined statistical information. This informa-

tion records how accurate a process is at determining the function of a cell based on the context in which that cell appears.

7.11.2 Structure

The task of determining the structural model of a table can be defined as follows:

> **task definition:**
> *For each cell in the table, determine the set of cells which can be reached from it via the navigation of the table.*

Heuristic Structure Determination

- Using functional and physical cues. This algorithm is split into two basic parts, those concerned with cells judged as being access cells, and those judged as being data cells. The algorithm is presented in Appendix E.

- Content and physical based structural determination. For this heuristic, a semantic net (WordNet) is used to suggest relationships between cells arranged in the table in columns.

- Any cell at the top which is not a source distributes over all the cells aligned below it.

- If you have a cell which is of a particular semantic type above a column, or to the left of a row, of cells of another type, then link it. One of the hardest problems in determining structure is to decide if a column (or row) with no spanning is a series of cells on a particular level of a category, or if the top cell in the column (or the left hand cell in the row) is in fact the superior cell. It is possible to write a series of rules based on simple syntactic patterns for semantic units in the table which can be triggered by these unit width patterns in the physical table, a simple example being an expression of unit (e.g. m^3) over a column of numbers.

7.11.3 Semantics: Relational

The task of determining the relational semantic model of a table can be defined as follows:

task definition:
Determine the set of categories, their inclusion in $\mathcal{CON}$ and $\mathcal{DIS}$ and the mapping from these sets to the members of $\mathcal{D}$.

Simple Relational Semantics Processor

The algorithm progresses in the following manner.

1. For each of the data cells in the table, generate the paths (vertical and horizontal) to those cells using the current model of the table's structure (T^{struc}).

2. Compute the dependencies (vertical and horizontal) for those paths.

3. Filter the dependencies. It is possible that the simple manner in which dependencies are computed results in multiple dependencies for a particular cell content. In this case, we progress as follows:

   - Prefer aligned dependencies to those which are not aligned.
   - If some of the cells are dependent on cells above and others to the left, then take the most common.

4. Compute the maximal dependencies.

5. Compute the categories.

6. Partition the dependent and independent categories.

7.11.4 Semantics: Inter-Cell Relationships

The task of determining the relational semantic model of a table can be defined as follows:

task definition:
For each cell, determine the cell contents and for each pair of cell contents determine which, if any, relation holds between them.

Simple ICR Processor

The algorithm used to determine the inter cell relationship has the following outline:

- The first step is to determine the cell contents for each node in the category tree.
- Once we have the set of contents, we form a matrix for the category allowing us to consider between which sets of contents we are willing to compute relationships.
- Determine the relationship.

Determining the relationship between the contents is the step in the processing of tables which requires some form of domain knowledge (Section 1.4). However, there is a strategy which allows for some approximation to be made using more general forms of knowledge. In many cases, the contents of cells can be described by certain semantic tags: dates, numbers, units of measure, etc. Additionally, information gleaned from the document, or at least from certain parts of the document in which the table appears can provide some amount of dynamic information such as the presence of noun groups and so on[1].

SIMICR is a simple module which looks for certain types of regular phrases in tables cells and assigns tags to them. It then uses heuristic techniques to guess what type of relationships these might be. The semantic tags are placed using a regular grammar encoding a number of general semantic units (see Appendix F).

In addition, the module uses the identification of certain words to trigger recognition of the type of contents the cell contains. For example, `type`, `class` and `members` are all words which often indicate that there is a sub-super type of relationship between the parent cell and its children.

7.11.5 System Output

The output of the system takes two basic forms. Firstly, there is the simple SGML marked up results of the modules. These files are used for getting at the results for purposes of performance measuring and so on. The second mode of output is the

[1]Noun groups and other chunks cannot normally be found in the short spans of text found in table cells due to the manner in which the stochastic processes used to detect them work.

HTML file. The modules are capable of reporting their results as an HTML file which includes the table either coloured to indicate the assertions made, or some other description of the result (e.g. the ICR results are displayed as a matrix indicating the relationship found between cell contents).

7.12 Chapter Summary

This chapter has described the architecture and main components of the Tabpro system. This does not constitute a thorough description from which the system can be implemented but an indication of what techniques were used to construct Tabpro and the style of engineering used. Further information on the commands for Tabpro can be found in Appendix D.

Chapter 8

Evaluating the TabPro System

This chapter presents and analyses results of running the Tabpro *system with a number of different scripts, in a number of different contexts for all of the model instantiation tasks outlined in the preceding chapters.*

8.1 Introduction

The evaluation follows the basic methodology used by the IE community: precision and recall. **Precision** refers to the proportion of answers which were correct. High precision indicates that the assertions made by the system are accurate. **Recall** is the proportion of the desired results which are actually asserted. Clearly, the goal is to attain high precision and recall. Costs and benefits generally lie in trading precision for recall. It should be noted that the evaluation data is not unseen, i.e. it was used as test and development data during the design of the system, and conseqently, the evaluation presented here can not be treated in the same manner as the strict precision and recall reports of formal evaluations.

The system is evaluated up to the point at which integration is expected to occur with a complete IE system (Section 1.4). As for the identification of inter cell relationships, an informal investigation into the quality of performance is given.

The basic approach to evaluation is essentially quantitative. However, there is good reason to supply qualitative analyses of the system's performance. There is definitely a modal table type (the matrix table) and consequently, results are going to reflect in some way the proportion of tables which can be described in this manner. Looking at the less frequent, sometimes unique, tables in the corpus, and the system's

ability to process them, will also give us some idea as to the completeness of the model and the generality of the system. Again, these levels of complexity and the nature of the model reflect the need for content based analysis, e.g. the example shown below in **Table (8.1)**.

(8.1)

Dickinson et al (1965)	Dam x sire High risk x High risk	Dam x sire High risk x low risk	Dam x sire Low risk x High risk	Dam x sire Low risk x low risk
Suffolk	41(19/46)	69 (48/69)	10 (4/38)	18 (5/27)
Dickinson et al (1965)	Scrapie x Scrapie	Scrapie x No scrapie	No scrapie x Scrapie	No scrapie x No scrapie
Commercial flocks, Suffolk	Not reported	66 (38/57)	12-71 (7-40/56)	Not reported
Experimental mating,Suffolk	95(20/21)	81(25/31)	Not clear	18 (5/27)
Dickinson et al(1974)	Scrapie x Scrapie	Scrapie x No scrapie	No scrapie x Scrapie	No scrapie x No scrapie
Suffolk/Blackface F1*	100 (8/8)	62 (20/32)	33(14/43)	30 (38/125)
Suffolk/Blackface F2*	77(10/13)	59(26/42)	45(10/22)	25(7/28)
Hourrigan et al(1979)	Scrapie x Scrapie	Scrapie x No scrapie	No scrapie x Scrapie	No scrapie x No scrapie
Suffolk	78 (14/18)	42 (13/31)	39(50/129)	25 (26/105)
Foster and Dickinson (1988)	Scrapie x Scrapie	Scrapie x No scrapie	No scrapie x Scrapie	No scrapie x No scrapie
Suffolk	99(274/277)	Not reported	56 (37/66)	Not reported

8.2 Cell Function Determination

This task determines if a given cell is an access cell or a data cell.

task definition: *For a table T assign each cell to at most one of $\mathcal{D}$ or $\mathcal{A}$.*

Understanding the results requires that the role and the later significance of the classified cells be taken into account. As the access cells are more important due to their distribution over the data cells, one way in which the results may be rationally weighted is to normalise the precision and recall values in proportion to the distribution of access and data cells. This results in an amplification of the difference between the system's performance in assigning the two functional classes to cells. The ratio of data cells to access cells is 2.53 according to the set of tables gathered and marked up for this evaluation exercise.

This ratio means that 72 percent of the cells are data cells. Consequently, a baseline for results is 72 %. This is the performance of the system if all cells are classified as data cells. Results weighted in this manner are given in summary form when the scripts are compared in Section 8.2.7.

The results represent the average values for the classifications. These averages can be calculated over either cells, tables, or documents . The main result presented in the analyses below are from the documents . Those values in parentheses (()) represent the results per cell and those in square brackets ([]) represent the results per table.

To calculate the scores per cell, the corpus is considered as a large set of cells (all the cells in all the tables in all the documents). To calculate the scores per table, a per cell score is calculated for each table, a total is computed for all the tables in the corpus and this value is normalised by the number of tables in the corpus. To calculate the values per document, the per table values are computed for each document, summed and then normalised by the number of documents in the corpus.

8.2.1 Naive Bayes Classification

The naive bayes classification module was described in Section 7.11.1. Evaluating this module requires an experimental methodology which divides the corpus into training and test data, then rotates the division so that the entire corpus can be

evaluated. 10 partitions were made, i.e. 10 training sets and 10 test sets. Before partition, the corpus was randomised so that no significance could be given to the order of document collection, or the particular content domains of the documents.

Local Physical Features

This experiment uses only the features describing the *local physical context of the cell.* These four features describe the context of the cell in terms of its top, bottom, left and right neighbours. The context can be one of one to one, one to many, many to one, internal space, undefined (edge of table). This experiment acts as a baseline for the hypothesis that table processing requires some level of inspection of the *content* of the table and the document.

```
run SIMFUN +context +local;
```

The type of errors made by this module with these parameters can be split into two classes. The first are those errors relating to the distribution and number of classifications, the number of input parameters and so on. This is simply the character of the naive bayes classification algorithm, and its tendency towards modal classification. The second set of errors are to do with ignorance of the content of cells and the relationship that content has to the document and the domain.

Looking at the physical context and not at the content will never allow the system to identify structure templates (Section A.2.4). Additionally, it is not very good at identifying spanned access cells of the vertical type, though horizontal parent child relationships seem to fair well (Section A.2.1). Other problems include embedded structure which is a pathological content problem and cut-ins.

In general, this method is likely to go for the most common form of structure, due to the simple statistical nature of the algorithm, and consequently will more often than not assign a matrix table arrangement of functional cells.

1

	DATA	ACCESS	*total*
recall	*96.94 (97.63)*	*80.51 (78.31)*	*91.32 (92.18)*
precision	*89.49 (91.60)*	*93.77 (93.65)*	*91.01 (92.08)*

Content Based Features

The use of content based features for classification can be parameterised by the following flags.

`nondependent` (default value: `true`): those features not dependent on document elements being present (e.g. section headings, table captions, etc.).

`dependent` (`false`): those features dependent on document elements.

`condition` (`false`): only include dependent features if the document element is present.

`strict_condition` (`false`): set dependent and condition to true.

Document Content Here, the flag `nondependent` was set to true (by default) and the flags `dependent` and `condition` were set to false. In other words, only the features relating to the text in the body of the document were used. Those features which required the presence of a particular document element were turned off.[1]

run SIMFUN *+content +document*;

This parameterisation is sensitive to any indication of a link between cell content and document content, marking such 'content' cells as access cells. The results are not too bad; however there is, of course, no regard for the location of the cell. Also, any cell which is not content related to the document simply comes out as a data cell (due to the high percentage of data cells compared to access cells).

2

	DATA	ACCESS	*total*
recall	*86.89 (89.59)*	*47.35 (45.17)*	*73.37 (77.07)*
precision	*74.77 (79.37)*	*69.37 (66.83)*	*73.20 (76.98)*

[1] In implementing the system, a general strategy was used in the use of flags indicated by arguments to the modules. If a feature of the system could be included or excluded then a '+' or '-' symbol was used. If the flag was unique than only the '-' symbol was used. During the term of the implementation, this strategy was not enforced and so if there are any apparent ambiguities or inconsistencies the description of the module and the experiment should be referred to.

Document Content only Tokens In this case, only those content features describing tokens were used, not noun groups or noun group heads.

```
run SIMFUN +discrete +content +document -noungroup -noungrouphead;
```

Tokens, supplying no semantic (or even syntactic) information, are not particularly useful candidates for functional determination as the results seem to suggest being worse than those for content in general.

3

	DATA	ACCESS	*total*
recall	*87.43 (89.77)*	*36.28 (35.77)*	*70.32 (74.55)*
precision	*71.77 (76.89)*	*65.01 (61.97)*	*70.15 (74.47)*

Document Content only NounGroup The same experiment, but with noun groups instead of tokens.

```
run SIMFUN +discrete +content +document -token -noungrouphead;
```

Some improvement over Result Table 3 is found when noun groups are used.

4

	DATA	ACCESS	*total*
recall	*87.17 (90.47)*	*39.58 (37.79)*	*70.98 (75.63)*
precision	*72.29 (77.56)*	*67.03 (65.21)*	*70.85 (75.54)*

Document Content only NounGroupHead This experiment completes the series varying over the type of content features.

```
run SIMFUN +discrete +content +document -token -noungroup;
```

The results are marginally better than those for noun groups (Result Table 4).

5

	DATA	ACCESS	*total*
recall	*86.75 (90.13)*	*41.64 (39.32)*	*71.47 (75.81)*
precision	*72.88 (77.89)*	*67.52 (65.16)*	*71.32 (75.73)*

Document Content Dependent Here, the dependent features are turned on. However, there is no conditional requirements that the document elements be present for the table at run time. This means that the absence of that document element is the same as the feature being false for that document element.

```
run SIMFUN +content +document +dependent;
```

6

	DATA	ACCESS	*total*
recall	*88.32 (90.75)*	*41.04 (40.58)*	*72.78 (76.61)*
precision	*73.76 (78.34)*	*69.09 (67.59)*	*72.59 (76.52)*

Document Content Strict Condition -NonDependent This run is the same as Result Table 6, however, the nondependent features are turned off.

```
run SIMFUN +content +document +strict_condition -nondependent;
```

7

	DATA	ACCESS	*total*
recall	*96.93 (97.37)*	*19.60 (19.93)*	*71.06 (75.54)*
precision	*69.42 (74.55)*	*89.94 (89.02)*	*70.83 (75.46)*

In summary, it seems that for content based features from specific document locations (*i.e.* those found when table captions, sentences referring to tables and section headings are present) offer more precision than content features in general. What this suggests about the nature of the text's interaction with the content of the table is that discussion about the table bears some relationship to the functional aspect of the table. The poorer precision for content based analysis from the document in general indicates that the tables are only one part of the document and in general precipitate focused description and discussion which doesn't pervade the document as a whole.

Content Based and Local Physical Features

In these experiments, a mixture of content based and local physical features were used.

Document Content Context Local This experiment combines the non-dependent content features with the local physical features.

```
run SIMFUN +content +document +context +local;
```

The results indicate that the non-dependent features pull down the performance of the physical analysis. This is in line with the observations made above about where in the document table sensible content can be found.

8

	DATA	ACCESS	*total*
recall	*89.50 (91.66)*	*79.33 (78.21)*	*85.35 (87.87)*
precision	*87.78 (90.98)*	*79.90 (79.40)*	*85.15 (87.77)*

Document Content Strict Condition -NonDependent Context Local In this case, the above features are combined with the local physical features.

```
run SIMFUN +content +document +strict_condition -nondependent +context +local;
```

Again, the results reflect the use of context sensitive content analysis: access cell recall is pulled up slightly (though there is some degradation in the precision).

9

	DATA	ACCESS	*total*
recall	*96.18 (96.94)*	*82.45 (79.97)*	*91.40 (92.16)*
precision	*90.17 (92.10)*	*92.28 (91.91)*	*91.09 (92.05)*

Tables with complex shared data cells (Section A.2.3) are poorly done. In these cases, local structural cues are too heavy. Numerical content cues are missed due to certain hardwired assumptions regarding cells containing numbers[2]. Sliced tables (Section A.1.4), like structural templates, are very hard to identify with these table setting techniques and are completely missed.

In summary, adding in the specific content based features to the bayesian classification system gives a slight improvement over the use of purely physical features.

[2]The predominance of numerical data in the data domain, not the access cells, leads to large errors when certain analyses are performed on the table as a whole (such as looking for repeated cell contents). Consequently, certain content based operations and comparisons with the content of the document as a whole are restricted to non-numerical cell contents only.

8.2.2 Pattern Based Classification

The pattern based algorithm for functional classification is described in Section 7.11.1. Again, due to the machine learning nature of this module, train and test data was divided from the rotated corpus just as for the naive bayes algorithm.

Depth 1

Patterns of maximum depth 1 from the origin cell were the first to be investigated. An average of 402 patterns were found in the 10 training corpora. In this experiment, the patterns are used to provide a boolean vote for each cell. This means that, for example, a cell will be marked as an access cell if there are one or more patterns which indicate this and no patterns which indicate that it may be a data cell.

run PATTERNFUN *-d1*;

The results are reasonable, certainly above the baseline, however they don't compare with the naive bayes classification for physical features (Result Table 1).

10

	DATA	ACCESS	*total*
recall	*85.89 (89.00)*	*78.34 (80.34)*	*82.51 (86.56)*
precision	*92.85 (95.64)*	*93.65 (94.38)*	*93.02 (95.31)*

Depth 1, Pattern Count

In this experiment, the same pattern base was used. However, the voting system recorded the actual number of patterns which indicated a certain functional description for the cell, and not simply a yes or no vote as in the boolean case. A decision was made according to the majority vote.

run PATTERNFUN *-d1 -pc*;

The results are slightly better than Result Table 10; recall has been traded for a slight drop in precision.

11

	DATA	ACCESS	*total*
recall	*90.68 (92.08)*	*78.93 (80.87)*	*85.71 (88.92)*
precision	*92.56 (95.41)*	*93.56 (94.31)*	*92.99 (95.12)*

Depth 1, Context Physical Functional Equal Filter, Pattern Counter

More control is afforded if patterns which have the same physical shape but different functional shape are eliminated.

```
run PATTERNFUN -d1 -pc -cpfef;
```

The tighter control over the patterns results in better performance for data cells, but a loss in access recall for a slight gain in precision.

12

	DATA	ACCESS	*total*
recall	*91.20 (92.22)*	*74.14 (76.42)*	*84.28 (87.77)*
precision	*90.87 (94.03)*	*94.09 (94.54)*	*91.92 (94.16)*

Depth 2

Patterns of maximum depth 2 were generated from the corpus. The average number of patterns per training corpus was 1080. The experimental results follow the same pattern of improvement as for the experiments for patterns with a depth of one. However, the values are all considerably better for access cells when compared with the naive bayes classifier for physical context features (Result Table 1).

Depth 2, Pattern Counter

```
run PATTERNFUN -d2 -pc;
```

13

	DATA	ACCESS	*total*
recall	*94.69 [88.33] (90.49)*	*88.78 [79.30] (73.33)*	*92.08 [84.45] (85.65)*
precision	*93.47 [94.84] (96.15)*	*93.13 [95.62] (95.67)*	*93.59 [93.84] (96.03)*

Depth 2, Context Physical Functional Equal Filter, Pattern Counter

```
run PATTERNFUN -d2 -pc -cpfef;
```

14

	DATA	ACCESS	*total*
recall	*94.01 [88.19] (88.13)*	*83.84 [76.56] (71.31)*	*90.25 [82.78] (83.39)*
precision	*91.57 [94.84] (96.05)*	*93.24 [95.63] (95.68)*	*92.28 [93.92] (95.96)*

In summary, this pattern based algorithm can outperform naive bayes for the classification of access cells. However, how the algorithm improves over increased pattern depths and other matching and voting algorithms remains as further work.

8.2.3 Heuristic Based Classification

The above experiments have been concerned with using statistical evidence from the corpus and the document being processed. This next set of experiments uses heuristics based on observations of the corpus.

Syntacto-semantic (9)

Certain common semantic units can be found in the text and these might be exploited by assuming their roles in the table. For example, units of measure in parentheses are often access cells.

```
run HEURISTICFUN 9 -h9pu;
```

The results below demonstrate that this is a very accurate heuristic though the recall is low.

15

	ACCESS	*total*
recall	*0.23*	*0.066*
precision	*100*	*100*

Repeated Strings implies Access Cells (5)

Due to the presence of recapitulated categories (Section A.1.1), repeated cell contents might be thought of as a strong indicator of access structure. However, the precision of this intuition is not borne out by the experiments.

```
run HEURISTICFUN 5;
```

16

	DATA	ACCESS
recall	*0*	*15.31*
precision	*100*	*26.05*

Repeated Strings Oriented and Spanned implies Access Cells (Recapitulation) (5 -h5recap)

A modification of the above experiment greatly improves matters. Repeated cells which are in the same orientation and which are spanned by another cell are a good indicator of access function.

```
run HEURISTICFUN 5 -h5recap;
```

17

	DATA	ACCESS
recall	*0*	*8.73*
precision	*100*	*96.34*

Left Margin (7)

Cells in the left margin, due to the notion of the stub and the class of document elements which the system works with are usually access cells.

```
run HEURISTICFUN 7;
```

18

	DATA	ACCESS	*total*
recall	*0*	*57.02*	*16.07*
precision	*100*	*94.55*	*94.55*

Top Margin (8)

Similarly, cells in the top margin are generally access cells.

```
run HEURISTICFUN 8;
```

19

	DATA	ACCESS
recall	*0*	*19.50*
precision	*100*	*94.68*

Top Margin and Top Free Cells (8 -h8fa)

A variation on the above also permits cells which are free above (have no cells above them). This reduces precision but increases recall.

```
run HEURISTICFUN 8 -h8fa;
```

20

	DATA	ACCESS
recall	*0*	*21.02*
precision	*100*	*91.47*

Left Margin (7) and Top Margin (8)

Combining the above identifies a reasonable number of access cells.

```
run HEURISTICFUN 7 8;
```

21

	DATA	ACCESS	*total*
recall	*0*	*72.10*	*20.32*
precision	*100*	*95.22*	*95.22*

Caption Noun Group Head

Looking at the caption to the table (if present) and selecting the noun group heads usually indicates something about the structure and function of the table. Noun group heads which appear in the caption, in this experiment, are marked as being access cells.

```
run CONTENTFUN +tcngh;
```

22

	ACCESS	*total*
recall	*4.95*	*1.39*
precision	*87.13*	*87.13*

Specialised Noun Group Heads

Certain noun group heads found in the text (in the table caption, sentences referring to tables and section headings) are used to determine the access cells in the table.

```
run CONTENTFUN +tcngh +tsngh +shng;
```

23

	ACCESS	*total*
recall	*19.16*	*5.40*
precision	*89.59*	*89.59*

Combining Heuristics

Combining the above with the heuristic for the top and left margin cells still gives reasonable precision.

```
run HEURISTICFUN 7 8;
run CONTENTFUN -h HeuristicFun +tcngh +tsngh +shngh;
```

24

	ACCESS	*total*
recall	*75.69*	*21.33*
precision	*93.67*	*93.67*

Adding further heuristics to deal with repeated cells and a heuristic for cells which span the entire table (6, cut-in cells) and to ensure that all the daughters of a spanning cell have the same functional type (2) increases the recall for the access cells. Note that the recall for the access cells already exceeds that for access cells based purely on the structural features.

```
run HEURISTICFUN 6 7 8 5 -h8fa -p First;
run HEURISTICFUN 2 -h HeuristicFun:First -p Second;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun:Second;
```

25

	ACCESS	*total*
recall	*88.04 [91.27]*	*24.81 [37.64]*
precision	*92.66 [92.47]*	*92.66 [92.47]*

Heuristics for setting a cell spanning access cells to an access cell (3) and setting cells with the same contents to the same function (4) give a final reasonable results for access cells.

```
run HEURISTICFUN 6 7 8 5 9 -h8fa -p First;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun:First -p Second;
run HEURISTICFUN 2 -h ContentFun:Second -p Third;
run HEURISTICFUN 3 -h HeuristicFun:Third -p Fourth;
run HEURISTICFUN 4 -h HeuristicFun:Fourth;
```

26

	ACCESS	*total*
recall	*88.64 [91.79]*	*24.98 [37.92]*
precision	*92.80 [92.56]*	*92.80 [92.56]*

8.2.4 Combination of Naive Bayes and Pattern Based

Naive bayesian classification and pattern based classification were combined in the following script.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
```

This combination is not particularly fruitful, giving results which are similar to both the context based naive bayes experiment and the pattern based experiment.

27

	DATA	ACCESS	*total*
recall	*95.98 (96.99)*	*80.26 (78.08)*	*90.60 (91.66)*
precision	*91.92 (94.91)*	*93.75 (93.63)*	*92.80 (94.60)*

8.2.5 Combination of Patterns and Structural Heuristics

Combining the structural heuristics and the pattern based classification provides access cell analysis which is better than the physical context base line.

```
run PATTERNFUN -d1 -pc -cpfef;
run HEURISTICFUN 2 3 4 6 7 8 -h PatternFun;
```

28

	DATA	ACCESS	*total*
recall	*91.21 (92.31)*	*84.61 (86.48)*	*87.72 (90.67)*
precision	*92.65 (95.42)*	*93.43 (93.67)*	*93.04 (94.94)*

8.2.6 Combining Naive Bayes, Heuristic and Pattern Based Classification

The following sets of experiments trace the course of the development of a script which maximises recall and precision scores for the corpus. The development of this script was not an exhaustive investigation into the 'script space' of the system. Rather, the above results were compared, and an informed decision was made about

the best combinations of modules. The possible 'script space' if far too big for an exhaustive analysis.

Firstly, a simple script was used incorporating pattern based determination, naive bayes classification and some heuristic analysis.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +content +document +strict_condition -nondependent +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 7 8 -h ReturningOfficerFun;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun;
```

-1.8cm

The results are already at least as good as those for the base line physical context experiment with some improvement in the identification of access cells.

29	DATA	ACCESS	*total*
recall	*95.03 [95.48] (96.16)*	*83.15 [86.56] (80.80)*	*90.93 [92.23] (91.83)*
precision	*92.93 [94.17] (95.62)*	*92.20 [92.04] (91.85)*	*93.04 [93.01] (94.66)*

The first iteration in the development of the script introduced the heuristic for normalising the type of a cell's daughters. If the majority are access then they are all marked as access cells; otherwise they are marked as data cells.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +content +document +strict_condition -nondependent +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 7 8 -h ReturningOfficerFun;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun;
run HEURISTICFUN 2 -h ContentFun;
```

This gives some improvement in the identification of access cells.

30

	DATA	ACCESS	*total*
recall	*95.09 [95.53] (96.19)*	*84.39 [87.96] (81.73)*	*91.26 [92.93] (92.12)*
precision	*92.73 [93.78] (95.45)*	*92.28 [92.11] (91.93)*	*92.92 [92.83] (94.54)*

The second step normalised the contiguous rectilinear areas. It identifies areas which contiguous data areas and re-classifies some of those cells and/or the surrounding cells to ensure that the area is rectilinear.

```
run PatternFun -d1 -pc;
run SimFun +discrete +content +document +strict_condition -nondependent +context +local;
run ReturningOfficerFun -h PatternFun -h SimFun -maj;
run HeuristicFun 7 8 -h ReturningOfficerFun;
run ContentFun +tsngh +tcngh +shngh -h HeuristicFun;
run HeuristicFun 2 -h ContentFun -p Daughters;
run HeuristicFun 1 -h HeuristicFun:Daughters;
```

This results in a small increase in precision with a slight loss of recall.

31

	DATA	ACCESS	*total*
recall	*93.43 [93.22] (94.36)*	*83.93 [87.44] (81.27)*	*89.94 [91.53] (90.67)*
precision	*92.86 [92.82] (95.33)*	*93.06 [92.59] (92.69)*	*93.16 [92.64] (94.65)*

Next, cells which have the same content as other cells are marked as being access cells if they are in alignment.

```
run PatternFun -d1 -pc;
run SimFun +discrete +content +document +strict_condition -nondependent +context +local;
run ReturningOfficerFun -h PatternFun -h SimFun -maj;
run HeuristicFun 7 8 -h ReturningOfficerFun;
run ContentFun +tsngh +tcngh +shngh -h HeuristicFun;
run HeuristicFun 2 -h ContentFun -p Daughters;
run HeuristicFun 1 -h HeuristicFun:Daughters -p Rectilinear;
run HeuristicFun 5 -h HeuristicFun:Rectilinear;
```

This improves the recall for access cells.

32

	DATA	ACCESS	*total*
recall	*93.34 [93.15] (94.29)*	*87.31 [90.68] (86.52)*	*90.85 [92.30] (92.10)*
precision	*93.35 [93.35] (95.70)*	*93.12 [92.75] (92.94)*	*93.50 [92.99] (94.95)*

Rearranging the order in which heuristics are performed can alter the output. Here the rectilinear heuristic is performed last.

```
run PatternFun -d1 -pc;
run SimFun +discrete +content +document +strict_condition -nondependent +context +local;
run ReturningOfficerFun -h PatternFun -h SimFun -maj;
run HeuristicFun 7 8 -h ReturningOfficerFun;
run ContentFun +tsngh +tcngh +shngh -h HeuristicFun;
run HeuristicFun 2 -h ContentFun -p Daughters;
run HeuristicFun 5 -h HeuristicFun:Daughters -p Recap;
run HeuristicFun 1 -h HeuristicFun:Recap;
```

-1.8cm

This slightly improves the overall performance.

33

	DATA	ACCESS	*total*
recall	*93.43 [93.22] (94.36)*	*87.31 [90.68] (86.52)*	*90.91 [92.35] (92.15)*
precision	*93.41 [93.37] (95.74)*	*93.28 [92.87] (92.94)*	*93.60 [93.05] (95.03)*

The next iteration moves the repeated cell heuristic to an earlier application and introduces the heuristic which sets a spanning cell to access if all the daughter cells are access cells.

```
run PatternFun -d1 -pc;
run SimFun +discrete +content +document +strict_condition -nondependent +context +local;
run ReturningOfficerFun -h PatternFun -h SimFun -maj;
run HeuristicFun 6 7 8 5 9 -h8fa -h ReturningOfficerFun;
run ContentFun +tsngh +tcngh +shngh -h HeuristicFun;
run HeuristicFun 2 -h ContentFun -p Daughters;
run HeuristicFun 3 -h HeuristicFun:Daughters -p Recap;
run HeuristicFun 1 -h1uf -h HeuristicFun:Recap -p Rectilinear;
```

Again, the results improve the overall performance, and — importantly — the access cell classification.

34

	DATA	ACCESS	*total*
recall	*94.68 [93.96] (95.20)*	*88.77 [92.11] (88.94)*	*92.08 [93.06] (93.44)*
precision	*93.47 [93.79] (95.80)*	*93.13 [92.74] (93.02)*	*93.59 [93.09] (95.04)*

Finally, the pattern based module is employed to collect patterns from the current classification of the table and fill in any the classification of any cells which have yet to receive a functional description.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +content +document +strict_condition -nondependent +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 6 7 8 5 9 -h8fa -h ReturningOfficerFun;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun;
run HEURISTICFUN 2 -h ContentFun -p Daughters;
run HEURISTICFUN 3 -h HeuristicFun:Daughters -p Recap;
run HEURISTICFUN 4 -h HeuristicFun:Recap -p Same;
run HEURISTICFUN 1 -h1uf -h HeuristicFun:Same -p Rectilinear;
run PATTERNFUN -h HeuristicFun:Rectilinear -d1 -j -r -u -pc -h HeuristicFun:Rectilinear;
```

The final results are as follows.

35

	DATA	ACCESS	*total*
recall	*95.89 [95.97] (96.71)*	*89.76 [92.59] (89.70)*	*93.40 [94.53] (94.74)*
precision	*92.99 [93.78] (95.56)*	*92.41 [91.85] (92.22)*	*93.10 [92.55] (94.64)*

In order to test the hypothesis that content is required for interpretation of the table, the above script was also run with the content dependent factors removed.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 6 7 8 -h8fa -h ReturningOfficerFun;
run HEURISTICFUN 2 -h HeuristicFun -p Daughters;
run HEURISTICFUN 3 -h HeuristicFun:Daughters -p Recap;
run HEURISTICFUN 1 -h1uf -h HeuristicFun:Recap -p Rectilinear;
run PATTERNFUN -d1 -j -r -u -pc -h HeuristicFun:Rectilinear -p Fill;
```

As can be seen by the results below, comparing the key indicator — the performance on the identification of access cells — with that for the content based script

indicates that content is an important factor when determining the functional nature of the table.

36

	DATA	ACCESS	*total*
recall	*96.86 [98.69] (97.59)*	*83.90 [87.43] (81.37)*	*92.51 [94.09] (93.02)*
precision	*91.09 [92.05] (92.62)*	*94.10 [93.70] (93.83)*	*92.21 [92.14] (92.91)*

8.2.7 Summary and Conclusions

Comparing the significant results by normalising for the access cell to data cell ratio gives the following results.

37

Experiment	*precision*	*recall*
Result Table 1	*92.56*	*85.16*
Result Table 35	*92.57*	*91.50*
Result Table 36	*93.25*	*87.57*

This suggests that the content based script provides a better analysis over the corpus than the script with no content based classification.

8.3 Table Structure Determination

The table structure task can be defined as follows.

task definition:
Determine the set of pairs of cells which are linked via the simple table relations and their restrictions.

The module HEURISTICSTRUC was tested for its performance. This module assumes that cells have already been assigned a functional description (access or data cell). In order to isolate the performance of the module, the correct classification of each cell was loaded in from the appropriate marked up corpus file using the module LOADFUN. Testing breaks down into those heuristics which deal with sink cells marked as access cells, and those which are marked as data cells. The functional description used as input to this set of experiments was the correct description for each table, loaded from the marked up corpus.

8.3.1 Access

Here, those heuristics which deal with access cells are investigated.

Vertical Access → Access

Firstly, heuristics which look for links between access cells and access cells position above them were tested. In this case, the sink access cells are plain cells with at most one cell adjacent physically on each face. This excludes cut-ins and repeated cut-ins.

```
run LoadFun;
run HeuristicStruc 0 -h00 -h LoadFun;
```

These results show the total precision and recall. Precision is high and recall (being the recall for the total table), is reasonable.

38

precision	*recall*
97.38 %	*16.90 %*

An analysis of the errors made resulted in the following identified problems.

Abbreviated spans (Section A.2.1) are missed. Such cells require identification and repairing before a general analysis can be carried out. Under-spans (Section A.2.3) are not linked. Spaces and missing cells in the table cause problems with searching upwards for an appropriate source. Again, this is a matter of the uniformity of table markup, and errors in the initial input. The rare case of right hand indexing (right hand stubs) causes confusion. This is also a problem of the identification of tables of the class the system is designed to deal with. Right hand indexing is rare. Templates of course cause problems.

Vertical Access → Access

In this case, the sink access cells are cut-in and repeated cut-in cells.

```
run LoadFun;
run HeuristicStruc 0 -h04 -h LoadFun;
```

This form of access to access link is not particularly common, though it is important. The results actually reflect the frequency of the type of cell.

39

precision	*recall*
97.39 %	*00.83 %*

Vertical Access → Access

In this case, the sink access cells are underspans.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h03 -h LoadFun;
```

Again, as in Result Table 39, this phenomenon is rare.

40

precision	*recall*
97.86 %	*00.09 %*

Lateral Access → Access

Here the sink access cell looks to the left for the first access cell. The source will generally be a cell in the right hand side of the stub access structure.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h05 -h LoadFun;
```

41

precision	*recall*
98.98 %	*1.56 %*

The main cause for error in this part of the algorithm is the identification of horizontally versus vertically oriented heads and stub interaction. For example, a cell in the stub-head may be either a 'label' for cells in the stub, or it may be a 'label' for cells to the right in the head. In general, the vertical analysis is given, however on the occasions when the horizontal analysis is required, some form of

deeper processing is required. If it is possible to identify the 'label' and the cells it may distribute over, then we might be able to do some semantic analysis (e.g. crystal recognition) and match this with domain knowledge about possible relationships. A heuristic which does this has been implemented. However as it would introduce domain dependent processes into this evaluation it is not investigated. Suffice it to say that certain exceptions like this can only be found through content analysis.

Summation of Access → Access links

The complete performance of HEURISTICSTRUC on links between access cells is found by running the following script.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h00 -h03 -h04 -h05 -h LoadFun;
```

Which gives the following results.

42

precision	*recall*
95.78 %	*19.45 %*

8.3.2 Data

Data cells are always sinks, and must always link to an access cell, not another data cell. Data cells are generally linked to the first access cell above and below.

Lateral Data → Data

A simple heuristic: look to the left for the first access cell, then, link to it.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h06 -h LoadFun;
```

43

precision	*recall*
95.49 %	*36.61 %*

Vertical Data → Data

In this case, a data sink looks above for a suitable access cell.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h07 -h LoadFun;
```

44

precision	*recall*
99.38 %	*35.00 %*

8.3.3 Complete Heuristic Approach

The complete results for the heuristic approach outlined above and implemented by the module HEURISTICSTRUC can be found by running the following script.

```
run LOADFUN;
run HEURISTICSTRUC 0 -h00 -h03 -h04 -h05 -h06 -h07 -h LoadFun;
```

45

precision	*recall*
98.11 %	*91.51 %*

In summary, the discovery of a structural model of a table via a heuristic based module which assumes prior functional classification of cells produces reasonable recall with high accuracy.

8.4 Table Relational Semantics Determination

The instantiation of a relational model for a table is a task described as follows.

task definition:
Determine the set of categories, their inclusion in $\mathcal{CON}$ and $\mathcal{DIS}$ and the mapping from these sets to the members of $\mathcal{D}$.

The module SIMRELSEM (describe in Section 7.11.3) was evaluated. SIMRELSEM assumes that there is both a functional description and a structural description of the table already in place. Consequently, the modules LOADFUN and LOADSTRUC were used to load in the descriptions of the table from the appropriate marked up corpus document. This allows the investigation of the performance of SIMRELSEM to isolate the algorithm from those performing other tasks (functional and structural model analysis).

The module was run over the entire corpus.

```
run LOADFUN;
run LOADSTRUC;
run SIMRELSEM;
```

Results for the relational tasks may given in four forms. The first is the perfect score (norm). This indicates how many of the results were perfectly correct and is the form given in this experiment reported below. The second (sub) indicates the proportion of the results which may be marked correctly as sub set of the correct categories; e.g. if there is a category containing the category path `a.b.c` and the system provides `a.b` then a positive result is recorded. The next is the super category which is similar but deals with super sets. Finally (combined) all these results are combined.

The reason for this complex reporting of results is that some applications might be content to find some category information even if it is not complete (for example, using tables to discover knowledge rather than information). The complex results are provided for the combined experiments reported in the next Sections.

46

precision	*recall*
93.21 %	*95.01 %*

Extracting the categories is only half of the relational analysis. The mapping from the sets of categories to the data cells ($\mathcal{D} \in T^{func}$) is required to complete an instance of the relational model of the table.

This can be evaluated by checking the category paths and data cell information in the output of the relational and structural analysis. Obviously, it is closely linked

with the performance of the other parts of the system. The results for this experiment are as follows.

47

precision	*recall*
92.58 %	*94.62 %*

8.5 Integrated Performance

The results above (Result Table 46) represent the system running on ideal input — i.e. perfect functional and structural analysis. Further experimentation is required to see how the system performs live over the corpus. This investigation can be broken ino two parts. Firstly the combination of functional and structural analysis, secondly the combination of structural and relational analysis, and thirdly the complete analysis combining functional, structural and relational modules.

8.5.1 Functional analysis and Structural Analysis

Firstly, the functional analysis and the structural analysis were combined.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +content +document +strict_condition -nondependent +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 6 7 8 5 9 -h8fa -h ReturningOfficerFun;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun;
run HEURISTICFUN 2 -h ContentFun -p Daughters;
run HEURISTICFUN 3 -h HeuristicFun:Daughters -p Recap;
run HEURISTICFUN 4 -h HeuristicFun:Recap -p Same;
run HEURISTICFUN 1 -h1uf -h HeuristicFun:Same -p Rectilinear;
run PATTERNFUN -h HeuristicFun:Rectilinear -d1 -j -r -u -pc -h HeuristicFun:Rectilinear -p last;
run HEURISTICSTRUC -h PatternFun:last;
```

-2.1cm

Given the errors in the previous investigations, we would expect 91.63 % precision and 85.20 % recall based on the combination of precision and recall statistics for the two isolated tasks. However, the results, below, indicate slightly lower values for

each. This suggests that the dependency on functional classification information for the structural analysis task is not a simple linear one and involves some form of complexity.

48

precision	*recall*
89.42 %	*82.15 %*

8.5.2 Structural Analysis and Relational Semantic Analysis

The next experiment assumed perfect input for the functional classification and looked at how the structural and relational semantic processes worked.

```
run LOADFUN;
run HEURISTICSTRUC -h LoadFun;
run SIMRELSEM -h HeuristicStruc;
```

Again, analysis of the prior results for the two tasks would suggest a linear combination of results with 91.45 % precision and 86.94 % recall. And again, the actual results show that the complexities of the dependencies between the model components result in some non-linear combination of errors.

49

	precision	*recall*
normal	*82.29 %*	*83.16 %*
sub	*91.64*	*92.42*
super	*87.54*	*88.02*
combined	*95.27*	*95.70*

8.5.3 Functional analysis, Structural Analysis, and Relational Semantic Analysis

Finally, the whole system was integrated.

```
run PATTERNFUN -d1 -pc;
run SIMFUN +discrete +content +document +strict_condition -nondependent +context +local;
run RETURNINGOFFICERFUN -h PatternFun -h SimFun -maj;
run HEURISTICFUN 6 7 8 5 9 -h8fa -h ReturningOfficerFun;
run CONTENTFUN +tsngh +tcngh +shngh -h HeuristicFun;
run HEURISTICFUN 2 -h ContentFun -p Daughters;
run HEURISTICFUN 3 -h HeuristicFun:Daughters -p Recap;
run HEURISTICFUN 4 -h HeuristicFun:Recap -p Same;
run HEURISTICFUN 1 -h1uf -h HeuristicFun:Same -p Rectilinear;
run PATTERNFUN -h HeuristicFun:Rectilinear -d1 -j -r -u -pc -h HeuristicFun:Rectilinear -p last;
run HEURISTICSTRUC -h PatternFun:last;
run SIMRELSEM -h HeuristicStruc;
```

If we consider the performance of the systems described above in conjunction we can look either at cases where the input to a system is 'perfect' (i.e. loaded from a marked up file) or cases where the system is running 'live'. Looking at the various cases and predicting the results, again, in a linear manner, gives us the expected outcomes described in the chart below. The live axis shows which experiments are considered as passing on their errors to the next analysis phase. The perfect axis shows which phases of analysis were assumed as being perfect, i.e. providing 100 % precision and recall.

	live		
Perfect	Functional Structural	Structural Relational	Functional Structural Relational
Functional		r=77.67, p=76.61	
Relational	r=78.05 p=83.05		
			r=81.21, p=85.14

As in the previous two 'live' experiments, the results demonstrate that the complexity of the model does not result in linear combinations of performance results. That is to say, the errors propagate across the model in a combinatorial manner as we can see when comparing the results of the final experiment with the predictions in the above chart.

	precision	*recall*
normal	*69.21 %*	*68.75 %*
sub	*82.75*	*81.21*
super	*77.42*	*77.22*
combined	*89.18*	*87.82*

8.6 Inter-Cell Relationships: Qualitative Analysis

The simple module implemented to derive inter-cell relationships between cell contents uses crystal identification techniques to find certain classes of text in the table's cells. It then makes a decision about the possible relationship that might exist between the contents. Currently, the contents which are compared are those within a category, though as stated elsewhere (Section 4.30) there are other parts of the reading path which need to be investigated for semantic relationships.

The strategy adopted by the module is to find the largest spanning crystal in a given piece of text. It makes no assumptions about tokenisation. Consequently, crystals are often found which are simply sub-strings of words. For example, many of the units of measure are single letters (e.g. g for grammes) and are consequently found within space and otherwise delimited words. The reason that tokenisation is not considered is to avoid any cases where the tokenisation strategy is somehow mismatched with the crystal being matched, or where there are errors in spacing between words (a common problem). However, the over-recognition of the crystal resource indicates that the issue needs to be taken care of for any ICR module which uses the crystal based content identification.

In the majority of simple cases, where units of measure, data and time expressions and monetary units, either in parentheses or otherwise, are found in cells in isolation, a reasonable attempt is made at describing the possible ICR. Additional success is found when the semantic net resource contains some form of relevant information, and when certain key words are found in certain dominant positions in a category.

In **Table (3.4)**, the following results were obtained.

T^{relsem} ={

< cat_0, [Animal Type, { $cell_0$ },

{

$<$ cat_1, [Dairy, { $cell_1$ } $\emptyset$]>,
$<$ cat_2, [Beef, { $cell_2$ } $\emptyset$]>,
$<$ cat_3, [Veal, { $cell_3$ } $\emptyset$]>,
$<$ cat_4,
[Swine, { $cell_4$ },
{
$<$ cat_5, [Growing Pig, { $cell_5$ }, $\emptyset$]>,
$<$ cat_6, [Mature Hog, { $cell_6$ }, $\emptyset$]>,
$<$ cat_7, [Sow & Litter, { $cell_7$ }, $\emptyset$]>,
$<$ cat_8, [Sheep, { $cell_8$ }, $\emptyset$]>,
$<$ cat_9, [Goat, { $cell_9$ }, $\emptyset$]>,
}
]
$<$ cat_{10}, [Poultry, { $cell_{10}$ },{ $<$ cat_{11}, [Layers, { $cell_{11}$ }, $\emptyset$]> } >
}
]
>
}

The relationship NOMINAL_SUPER_TYPE is discovered through the structure due to the fact that the text type is found in the root of the category.

T^{relsem} ={
$<$ $cat_1$2, [Manure production, { $cell_{12}$ },
{
$<$ cat_{13}, [Tons/yr, { $cell_{13}$ } $\emptyset$]>,
$<$ cat_{14}, [Gallons/yr, { $cell_{14}$ } $\emptyset$]>
}
]
>
}

Identifying the spans of text Tons/yr and Gallons/yr as being units of measure,

the relationship UNIT_OF_MEASURE was asserted.

However, as this module is beyond the level of analysis stated for this research (Section 1.4), though still an important part of the overall table model, the technology is still in its infancy — the module really exists as a marker filling out the process.

There are still a number of directions to be explored for domain independent techniques for identifying inter-cell relationships. In general, these would require analysis of documents for textual evidence of certain ontological relationships between identifiable domain constituents (e.g. explicit statements such as `New York is a city`, or implicit statements such as `the city of New York`).

8.7 Summary of Performance

In judging the performance of the system it is perhaps unclear what should be used as an objective measure. The performance as described above is separated into a number of subtasks, as well as a complete system. In judging both the subtasks and the complete system, either systems performing similar tasks or the performance of humans may be used as a comparative measure.

Perhaps the only related task found in other research domains is the crystal identification task performed by the SIMICR module. This bears some similarities to tasks performed by general information extraction systems, particularly the named entity task defined by the MUC conferences. However, as only a qualitative account is given here, it is not particularly fruitful to pursue an in depth comparison.

The little corpus-based table related research focuses on tasks at the periphery of the research reported here, or on different classes of complex document elements going by the name 'table'.

Preliminary work presented in [HD97] describes similar work on a slightly different model of tables. A small corpus of tables was marked up to identify **domains** (a structural model of categories). Domains were identified through a number of typographic effects which were interpreted between tables in terms of *cohesion*. For example, the assumption that siblings are of not only similar semantic type but also, and as a result, similar typographic description allows a system to group cells exhibiting typographic similarities and to identify a 'label', if present, for that set. Templates were used to restrict the possible space of physical configurations (much

like the prototypical patterns described in Section A.2).

The task, as translated into the terminology of this thesis, was that of structural analysis. The results, for unseen data, were 54 % precision and recall. This is probably best compared with the combined task of functional and structural analysis, which here scored 89.42 % precision and 82.15 % recall, though for the evaluation presented in this thesis, the data was not unseen as in the previous study.

With reference to the criteria listed in the previous chapter (page 169) it can be claimed that: the system responded in a reasonable time (2), is capable of providing a measure of the confidence in its results via the design of the hypothesis record, though this mechanism was not fully implemented (3), is reasonably robust to input errors via the implementation of a script for translating LaTeX and HTML into the system XML (4) and degrades in a reasonable manner as an artifact of the quasi-blackboard design (5).

8.8 Chapter Summary

This chapter has presented an evaluation of the TabPro system in terms of the precision and recall measures commonly used by the IE community. Although the results are not presented in a comparative manner, due to the lack of suitable comparable material, certain subtasks undertaken by the system represent a significant and clear improvement over exploratory work done before the work in this thesis was started ([HD97]).

The implementation of the TabPro system uses the model of tables at all times to describe its progress, as hypotheses about the current table. Its results also suggest something about the model in terms of the relationships between levels of the model by the propagation of errors through the levels of analysis. Though these effects are not simply a matter of the model and involve the performance of the underlying analysis techniques (machine learning classification techniques, general knowledge bases, chunking, crystal detection, etc.) which cannot be separated from each other, it does suggest that, due to the complex nature of the propagation of errors, the table is a highly ambiguous form of information presentation. In fact, looking at the process in reverse, the ambiguities get greater and greater as the manner in which the information expressed becomes more and more restricted until the final graphical record of the information is placed in the page.

This in itself is great motivation for considering any form of table processing in as broad a context as possible both in terms of the context of the object being analysed (i.e. including the document which includes the table), and the possible resources and techniques being employed (NLP, general IE techniques, etc.).

Chapter 9

Conclusions and Appraisal

This chapter highlights the achievements resulting from the research reported in this thesis. It then presents a critical appraisal of this research in order to position these achievements and their significance in the table processing field. Further work is discussed and some final conclusions are drawn.

9.1 Contribution

The main achievements of this research can be seen as the results of a simple goal: to integrate tables into the document types acceptable to information extraction systems. Although designing and developing such a system is in itself a considerable task, the resources developed in order to get to that stage represent the real contribution of the research. In a general sense, this is an exploration of the table, a task which demonstrates the often hidden or ignored complexities within its highly ambiguous presentational formats.

The work reported in this thesis has demonstrated

Tables appear in many documents	Section 1.2
and contain much desirable information.	Section 1.1
Prior work in tables	Chapter 2, Section 3.1
has failed to provide a complete table model.	Section 2.5
A model of tables.	Part II
A system capable of constructing	

instances of that model.	Part III
Content is important for the analysis of structural document elements.	Chapter 8

9.2 A Critical Appraisal of the Thesis

This thesis presents two major results. The first is the development and presentation of a model of tables suitable for diverse applications. The second is an application of that model to the task of information extraction. In developing the model and applying it to the IE task, terminology for describing the space of tables has also been presented.

In developing the model, many applications and other areas of research concerning tables were considered. However, the model has only been applied to the IE task and a complete evaluation of the model as a general resource has not been supplied. Compounding this problem, and as a result of the lack of any prior art in the area, the evaluation of the T/IE system also lacks a setting in established work and as such is required to present both an evaluation methodology as well as an evaluation of the actual system. To make matters worse, the corpus over which the system is evaluated is not an arbitrated and standard resource but one constructed within the project as a whole.[1]

In summary, the work reported in this thesis can reasonably be criticised for being too self contained and essentially providing both the data and the measure. However, viewed in a more positive light, it can be said that this thesis summarised the field of IE from tables and provided an initial point of comparison. It

- Defines the table and presents the development of a model which underlines the complexities of the table (Part II).
- Identifies how this model of tables should be integrated with conventional IE systems (Chapter 1, Chapter 3).

[1]Other, recent papers on certain table processing tasks have reported the use of certain corpora, however these tasks are not as complex as that reported here and consequently those corpora are not suitable for training and evaluating a complex T/IE system.

- Employs the model to establish a number of tasks which may be carried out by a table processing system, and which build to a high level goal (Part II, Part III).
- Defines evaluation techniques for each of those tasks (Part 8).
- Provides a DTD capable of describing a corpus of documents which may be used for development and evaluation purposes (Chapter 6).
- Describes the design and implementation of a system and its evaluation with respect to the tasks (Chapter 7, Chapter 8).

The research reported here makes great use of a corpus of examples. All the examples reported in this work are taken from either the corpus used for the development, training and evaluation of the T/IE system, or from a more adhoc collection of paper based examples which have been collected throughout the course of this work and before in other projects. When employing examples for the development of the model, the attitude was always to find examples which offer new local and global phenomena that might test the current iteration of development, or require modifications to it. Such a strategy is good for broadening the coverage of the model, however it doesn't capture anything of the distribution of phenomena over the space of tables.

As a consequence of this, the evaluation of the system over the corpus can only say something about its performance over examples of the phenomena used in developing the model. In other words, the evaluation makes a statement about the system's ability to process tables which can be described by the model. The results do not represent the performance of the system over, for example, a representative set of documents from a particular domain — an experiment which would say something about the distribution of phenomena in the space of tables examples. It is expected that, as with any other AI problem, the majority of cases are those which can be adequately dealt with and modeled, and the harder cases require greater effort for an increasingly complex and small subset of table examples. The evaluation, then, is in some manner proportional to the performance of the system over a randomly selected, or representative domain corpus.

The model presented in this thesis is a general model of tables for information processing systems. However, only the task of information extraction has been discussed at any length as a validation of the model. In order to validate the claim that

the model is general to any table processing task, implementations and evaluations of those tasks are required that employ the model. Applications which are considered to be table processing tasks include:

Table identification in document images: given a document image, discover the areas of the image which should be processed as tables (e.g. [SP92]).

Table identification in free text: given a document encoded in a basic format (e.g. Unicode or ASCII), discover the tables (e.g. [KD98]).

Table identification in tagged text: given a document marked up to some extent with tags (e.g. XML or SGML style tags), locate the tables.

Table markup in free text: given a table in free, unmarkedup text, markup the table with cells as per the physical model of the table presented in this thesis (e.g. [KD98]).

Table markup in tagged text: given a table markedup to some extent with tags, markup the table with cells as per the physical model of the table.

Table translation: given a table described in one format, produce the same table described using a different system (e.g. HTML to ASCII).

Document Retrieval based on or including information from tables: provide a system which can make sense of queries in terms of information presented in tables in the document base.

Information Retrieval based on or including information from tables: provide a system which can make sense of queries in terms of information presented in tables in the document base (e.g. [PC97]).

Text Mining based on or including information from tables: integrate table sensitivity to text mining ([Hea99]) applications.

Information Extraction based on or including information from tables: integrate table processing abilities to the IE task (e.g. [DHQ95], [HD97]).

Knowledge Extraction based on or including information from tables: make more general use of the domain knowledge displayed by tables.

Table Generation from non-linguistic data: transform some non-linguistic data source into a human readable table.

Table Generation from non-linguistic data to be incorporated in another document: transform non-linguistic data into a table which is appropriate for a complete document either human generated or automatically generated.

Table Editing: provide a system capable of manipulating a table at all levels of the model in a consistent manner (e.g. [BEF84], [Wan96])).

Table Formatting: produce a table consistent with certain formatting rules (e.g. [Wan96]).

Table Rendering: create a table image for a document.

Table Checking: ensure certain stylistic conventions are obeyed.

Table Verification: ensuring that the information presented in a table is in some way consistent with that in another data source.

9.3 Further Work

Throughout this thesis, a number of pointers have been placed indicating areas for *further work*. Additionally, Section 9.2 above suggests a spectrum of applications for the table model. The pointers to further work are summarised below.

- How the table fits into the discourse structure of the document both as an object to be referred to as well as in terms of its content and how it relates to the flow of the document narrative (page 47).
- Integration of the table processing system with a full-scale information extraction system (page 61).
- The encoding of more variation in the physical model of the table, for example the orientation of text, fonts, colours etc. (page 75).
- Further exploitation of the document text for information regarding the domain (page 118).

- An investigation into the phenomena in the document which introduce new relationships between elements in the table (page 124).
- Exploitation of context dependency between data cells (page 124).
- Understanding of tables which employ the fracturing of linguistic wholes and distribute those elements through a number of related cells (page 124).
- Implementing an algorithm which can identify the class of tables described in this thesis (page 169).
- The implementation and testing of further machine learning based algorithms for certain table processing sub-tasks (page 202).
- Algorithms for identifying domain dependent access cell orientation cues (page 215).
- The implementation of high quality crystal recognition algorithms (page 221).

9.4 Conclusion

As the section on further work suggests, the fledgling field of table processing will support a great amount of research in the future. It is hoped that this thesis has helped in some way to establish what the state of the art is, what tables might actually be and has identified where ambiguities lie in order that table processing systems might concentrate their efforts in those areas on the route to increasing accuracy and the overall quality of performance.

If there is a single lesson to be drawn from this work, it might be that tables inhabit a reasonably simple physical space which can be described accurately in a transparent manner. However, establishing the meaning of the table by automatic methods is a complex problem, a factor which is demonstrated by the increasingly complex tables which can be found with a little effort in generally available documents, and which pose few problems for the human reader. The implication of this broad array of complexity, from complex meaning to simple physical description, is that table processing requires a computational model which utilises information flow in both directions of a pipeline between tasks dealing with the physical aspects of tables, and for those dealing with the logical aspects of tables.

Summary of Part III

The final part of this thesis has described the development, implementation and analysis of a table processing system suitable for the IE task. In doing this, a number of subsequent achievements were made including defining subtasks for the system and evaluation strategies for the results. The final chapter summarised the thesis and suggested areas of further work for the table processing field in general and for applications of the model presented in this thesis in particular.

Appendix A

Organisation and Restriction and Rendering Structure in Tables

This Appendix presents and discusses phenomena resulting from the organisation of the table and the restriction of the two-dimensional page. The first section details the phenomena, and the second provides an analysis with respect to the model of tables presented in this thesis.

A.1 Organisation and Restriction

Due to the limitations of the two dimensional page, the author's hand is often forced. Some noticeable effects of this restriction are

- Recapitulation
- Table Partitioning
- Over-spanned labels
- Slicing

which are discussed in the following sections.

A.1.1 Recapitulation

If the author decides that a particular cell can be characterised in two or more independent dimensions (for example, `animal` may be specialised as `horse` and `pig`, as well as `old` and young) and reorientation of the relevant categories is not available

due to physical restrictions, then recapitulation occurs. This is the multiplying out of a category into cells with repeated contents.

(!A.1)

Animal			
Horse		Pig	
old	young	old	young
v_0	v_1	v_2	v_3

Recapitulation can be removed if it is possible to reorientate the structures involved.

(!A.2)

	Animal	
	Horse	Pig
old	v_0	v_2
young	v_1	v_3

Clearly, such organisational issues have economic impact: 9 cells versus 11 cells.

Partial recapitulation can also occur when certain (sub-)categories are not known or not appropriate, as shown in this example from (P33), in which `Recognition rate (%)` is a member of a category which is partially recapitulated:

(A.3)

		Number of training words				
		0	4		10	
Speaker	Adaption method	Recognition rate(%)	Recognition rate(%)	*Error reduction rate(%)*	Recognition rate(%)	*Error reduction rate(%)*

From this, we might hypothesise that any structure in a table may be modeled as some form of partial recapitulation. In the extreme case there is no commonality at all between instances of recapitulated domains. The issue becomes a matter of understanding *why* certain cells in recapitulated domains are *not* present. In the extreme case, the reason is likely to be that there is some restriction, in effect a semantic relationship, acting between cells. Recapitulation indicates that the category being recapitulated is in some way *independent* from or *orthogonal* to the domain below whose values it appears. However, this does not necessarily mean that it doesn't restrict the meaning of the superior cells.

In the following (**Table (A.4)**, which presents only the stub of a table complete with the related cut-in cells — those cells which span across the entire table), the category of `values` is recapitulated below the cut-in, however the final two value rows, below the contracted cut-in `average` are better considered as being from a different category as there is only one member in common with the other members of the recapitulated category.

(A.4)

value
Usage of the English Articles(140 sentences, 380 nouns)
correct
reasonable
partially correct
incorrect
% of correct
The Old Man with a Wen(104 sentences, 267 nouns)
correct
reasonable
partially correct
incorrect
% of correct
clan essay "TENSEI JINGO"(23 sentences, 98 nouns)
correct
reasonable
partially correct
incorrect
% of correct

average
% of appearance
% of correct

In addition, the following features of recapitulated categories can been observed.

- a recapitulated category must have more than one member.
- the repeated instances of the category must be contiguous structurally and not separated by other domains. i.e. the category must be either horizontally or vertically aligned and not 'interrupted' by cells from another category.
- the category must have equivalent immediate structure: i.e. all examples of the category must have a parent if at least one of them does.

A.1.2 Independent Partitions

In some cases (for example, the table in Figure A.1), a table is constructed which combines independent elements apparently in the same relational structure. The NASA table (Figure A.1, page 238), for example, cannot have a `flight` number for a Russian/USSR mission *and* a `flight` number for United States mission on the same row.

This organisation results in a number of empty cells. These can be removed if reorientation is applied:

Mission	Crew	United States						Russia/USSR		
		All			Shuttle					
		Flight	People	Trips	Flight	People	Trips	Flight	People	Trips
Through 1990		69	162/10	270	16	119/10	199/16	72	85/2	152/3
1991										
STS-37	5	70	165/11	275/17	39	122/11	204/17			
STS-39	7	71	170/11	282/17	40	127/11	211/17			
Soyuz TM12	3							73	88/3	155/4

Figure A.1: A Complex Table

(A.5)

Mission	Crew			Flight	People	Trips
Through 1990		US	All	69	162/10	270
			Shuttle	16	119/10	199/16
		Russia/USSR		72	85/2	152/3
1991						
STS-37	5	US	All	70	165/11	275/17
			Shuttle	39	122/10	204/17
STS-39	7	US	All	70	170/11	282/17
			Shuttle	7	170/11	282/17
Soyuz TM12	3	Russia/USSR		73	88/3	155/4

The above observations should be considered with respect to the notion of data dependency discussed in Section 4.5.2. The table above exhibits some very interesting and unusual features. Firstly, the second column (`Crew`) contains data cells (the number of crew on a particular mission). However, the complex column following this to the right includes some indexing information. This in some sense breaks up the contiguous rectilinear area of the table containing the data with a complex subset of access cells. As stated elsewhere, there are no hard and fast rules governing the production of tables, however, it would seem undesirable to break up the data with internal access cell complexes in this manner.

A.1.3 Over-Spanned Labels

When the physical rendering of a category is reoriented, labeling[1] may not be obvious. In the following example, the label `States` has a strict interpretation as the supertype of p and q. However, as p and q have had their (recapitulated) children reoriented the label `States` over-spans. In other words, the cell containing the text which represents the parent to a set of cells spans more material in the table than its siblings.

[1] A 'label' is not a well defined term, however here it is used to mean a parent of a set of sibling cells.

(A.6)

	States	ϵ	b
q	sequence	q	qq
	probability	1.0	.2
r	sequence	r	qr
	probability	0.0	.1

A.1.4 Slicing

Slicing occurs when the table is sliced in the vertical dimension and the cut off piece or pieces are laid out on the page in a sequence to the right of the original table section. The interpretations are equivalent as the following example illustrates (**Table (A.7)**). The parents L1 and L2 are places above two sets of cells containing their children. In the second horizontal rendering of this table, the vertical arrangement of the table is sliced and appended to the right with the parents still dominating the children logically even though there is no longer any alignment.

(!A.7)

L1	L2
V1	V2
V7	V8
V13	V14
V3	V4
V9	V10
V15	V16
V5	V6
V11	V12
V17	V18

→

L1	L2				
V1	V2	V7	V8	V13	V14
V3	V4	V9	V10	V15	V16
V5	V6	V11	V12	V17	V18

This effect can also occur with repeated headings.

(!A.8)

L1	L2	L1	L2	L1	L2
V1	V2	V7	V8	V13	V14
V3	V4	V9	V10	V15	V16
V5	V6	V11	V12	V17	V18

A.2 Rendering Structure In Tables

As a mechanism for the presentation of information, the table has a rather limited vocabulary of physical arrangements with which to organise its elements. Significantly, in addition to the alignment of cells described earlier, cells can be:

- **adjacent** to one another.

- **spanning** a group of cells.

In terms of the orientation of the structure in cells, there are only two dimensions to work with: structures will either be vertical or horizontal. These factors (the spanning and adjacency, and the orientation) are used to indicate groupings of cells and order in the reading paths.

One of the key factors governing the layout of the table and its relationships with the more abstract components of the table model, especially the structural component, is the opportunistic manner in which the physical table is derived. Once a certain amount of complexity exists in the table, in terms of the number of categories and the depth of those categories, the decisions made by the author regarding the placement of access categories and the access component of data categories if it exists, is dependent on the material already present in the table. For example, if there are two categories in the head and both require some form of structure (i.e. a 'label'), then at least one must have a horizontal orientation. This is illustrated by the first example (**Table (A.9)**) below:

(!A.9)

	$Label_0$			
	$Value_{0,0}$		$Value_{0,1}$	
$Label_1$	$Value_{1,0}$	$Value_{1,1}$	$Value_{1,0}$	$Value_{1,1}$

If the material in the stub requires a 'label' then this may be placed to the left of its children (i.e. a rotation of the vertical structure in the head) providing horizontal structure.

(!A.10)

		$Label_0$			
		$Value_{0,0}$		$Value_{0,1}$	
	$Label_1$	$Value_{1,0}$	$Value_{1,1}$	$Value_{1,0}$	$Value_{1,1}$
$Label_2$					

However, if it doesn't require a 'label' then it may appear in a single column as in the second example (**Table (A.11)**):

(!A.11)

	$Label_0$			
	$Value_{0,0}$		$Value_{0,1}$	
$Label_1$	$Value_{1,0}$	$Value_{1,1}$	$Value_{1,0}$	$Value_{1,1}$
$Value_{2,0}$				
$Value_{2,1}$				

This situation can be compared with the case when the second category in the head doesn't require a label (**Table (A.12)**), and when the category in the stub does (**Table (A.13)**):

(!A.12)

	$Label_0$			
	$Value_{0,0}$		$Value_{0,1}$	
	$Value_{1,0}$	$Value_{1,1}$	$Value_{1,0}$	$Value_{1,1}$
$Value_{2,0}$				
$Value_{2,1}$				

(!A.13)

	$Label_0$			
	$Value_{0,0}$		$Value_{0,1}$	
$Label_2$	$Value_{1,0}$	$Value_{1,1}$	$Value_{1,0}$	$Value_{1,1}$
$Value_{2,0}$				
$Value_{2,1}$				

In summary, it is the opportunisitc manner in which tables are constructed that causes much of the ambiguity found in the physical and structural table.

A.2.1 Grouping

The grouping of cells is indicated by one of the following mechanisms:

- **unlabeled cell group**: in this case the group must be indicated through proximity and alignment:

1. | v_0 | v_1 | v_2 | e.g. | 1990 | 1991 | 1992 |

2. | v_0 | | Thatcher |
   | v_1 | e.g. | Major |
   | v_2 | | Blair |

- **labeled cell group**: here a cell adjacent to one or all of the grouped cells in some way labels the cells:

  - **label-spanned cell groups**:

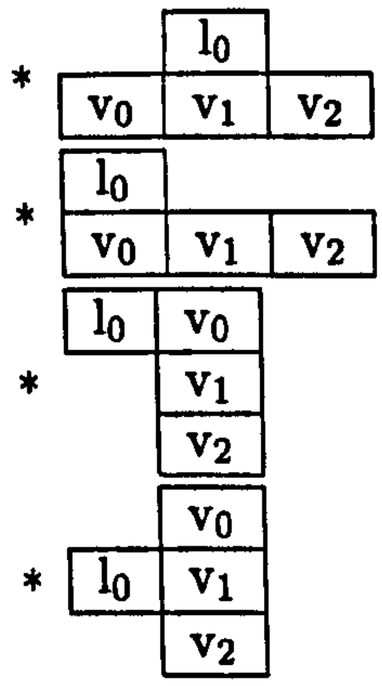
    * vertical | l_0 | | |
      | v_0 | v_1 | v_2 |
      e.g. | Years | | |
      | 1990 | 1991 | 1992 |

    * horizontal | l_0 | v_0 |
      | | v_1 |
      | | v_2 |
      e.g. | PM | Thatcher |
      | | Major |
      | | Blair |

  - **distributed label**: | l_0 |
    | v_0 |
    | v_1 |
    | v_2 |
    e.g. | Term |
    | 1 |
    | 2 |
    | 3 |

In addition, certain alternative indications of spanning may be encountered. These derive from discrepancies in the markup of the physical table (a problem that is avoided if the table were marked up according to structure). The following such variations have been observed.

- labeled cell group:
  - label-spanned cell groups:



Just as the normal physical representation of spanning may be confused with such phenomena as described in the following section on interruptions, so too can these exceptional forms.

A final class of variations, and one favoured by certain domains of a financial nature (e.g. SEC filings), is the indented distribution. The cell structure is generally something like the following.

(A.14)

Cost and expenses:
Interest and dividends
Foreign exchange loss, net
Other

This form is used both in cut-in like instances in which the dominant cell spans the entire table, and also in local structures in the stub. It is not entirely clear how this should be analysed. It could be a matter of justification, or it could be a matter of cell structure.

A.2.2 Interruption

The above is the standard repertoire of representing structure. Two other cases, collectively termed **interruption**, must be considered

- cut-in.
- substitution.

Cut-in The cut-in manifests on a single row (or column) of the table. It is a span taking up that row.

(!A.15)

l_0		
v_0	v_1	v_2
v_3	v_4	v_5
ci_0		
v_6	v_7	v_8
v_9	v_{10}	v_{11}

A cut-in has two cases:

- **precedented:** the cut-in cell is related homogeneously to a previous cell in the head or stub (not a previous cut-in) and indicates a change of context. This has some similarities with the physical template.
- **unprecedented:** the cut-in cell is simply an interruption of the structure of the table, and can be considered similar to a rotation of a spanning. In this case, it may still be related heterogeneously to cells in the head or stub or it may not.

precedented : which uses the cut-in to describe the stub (*cf.* the title which can appear in the same position at the top of the table but is used to describe the matrix.)

(A.16)

Unchanged options	
`l`	Left adjusted column
`c`	Centered adjusted column
`r`	Right adjusted column
`p{width}`	Equivalent to \ `parbox[t]{width}`.
`{decl.}`	Suppresses inter-column space and inserts *decl.* instead
Changed options	
`|`	Defines a column of width *width*. Every entry will will be enlarged by the width of the line in contrast to the original definitions of LaTeX
New options	
⋮	

unprecedented, unrelated :

(A.17)

value
Usage of the English Articles(140 sentences, 380 nouns)
correct
reasonable
partially correct
incorrect
% of correct
The Old Man with a Wen(104 sentences, 267 nouns)
correct
reasonable
partially correct
incorrect
% of correct
clan essay "TENSEI JINGO"(23 sentences, 98 nouns)
correct
reasonable
partially correct
incorrect
% of correct

average
% of appearance
% of correct

unprecedented, related :

(A.18)

Animal Type
Dairy
Beef
Veal
Swine
Growing Pig
Mature Hog
Sow & litter
Sheep
Goat
Poultry
Layers

More complexity is afforded by the cut-in head. A cut-in head is essentially the same as the cell described above but is a complex with internal structure.

In terms of reading path, the cut-in presents the following interesting problem:

- how should the cut-in be attached to the normal reading paths, and, subsequently, how should it access the following structure 'below' it in the table?

It may be appropriate to consider two categories of cut-in:

- Semantically related to previous material.
- Not semantically related to previous material.

In other words, is the cut-in cell a new category? If so, then we must consider how it is related to the subsequent material. In Figure A.1 the cell '1991' would still have the same interpretation if it either spanned the cells it scopes vertically on the left hand side, or if it appeared as a series of values for an unlabeled category listed, for example, on the left hand side. This may be modeled, then, as {(1991, STS-37), (1991, STS-39), (1991, Soyuz TM12)}. As for its relationship to previous material, it would seem appropriate to consider it as the first cell in a new reading path: i.e. it has no STR relationships with previous cells. The analysis above is also equivalent to transforming the cut-in cell to the arrangement termed an orthogonal domain.

Note that in this example case, there is no apparent hierarchical relationship between the contents of the cut-in and the contents of the cells immediately below it, however, there is some form of relationship between the cut-in and the cells in the category on the left hand side in that all of the missions took place in 1991.

This form of cut-in is perhaps the most simple. A cut-in which has a complex structure cannot be considered in the same manner as that presented here as it will

have some relationship to the cells below it in the table, and may not be simply interpreted as a rotation of a span which might be placed to the left as in this example.

We have already discussed how we might deal with the cut-in cell in Figure A.1. Figure A.21 (P18:1) offers another example. Here we can see that the cell `Swine` refers to the different classifications of swine (`Growing pig`, `Mature hog`, `Sow & litter`). Note that semantic knowledge is required to recognise when this classification stops and the table reverts back to individual entries (`Sheep`, `Goat`) before introducing another cut-in (`Poultry`).

Repeated Access Structures A phenomenon related to the cut-in is the repeated access structure. In this case, the head is duplicated in the body of the table in relationship to the hierarchical organisation of the access cells in the stub.

(!A.19)

l_0	$l_{2,0}$	$l_{2,1}$
$l_{0,0}$	v_0	v_1
$l_{0,1}$	v_2	v_3
l_1	$l_{2,0}$	$l_{2,1}$
$l_{1,0}$	v_4	v_5
$l_{1,1}$	v_6	v_7

The above demonstrates a vertical arrangement where the label adjacent to the repeated cut-in distributes over the cells below. The following variation indicates a lateral arrangement in which the the cell adjacent to the repeated material is distributed across the repeated cells.

(!A.20)

l_0	$l_{3,0}$	$l_{3,1}$
$l_{1,0}$	v_0	v_1
$l_{1,1}$	v_2	v_3
l_0	$l_{4,0}$	$l_{4,1}$
$l_{2,0}$	v_4	v_5
$l_{2,1}$	v_6	v_7

Substitution A substitution is a cell which, as a one off, appears in the place of a cell, or cells, of a different type.

(!A.21)

l_0		
v_0	v_1	v_2
v_3	v_4	v_5
v_6	s_0	
v_9	v_{10}	v_{11}

Animal Type	Manure Production		Percent Solids	Nutrient Content					
	Tons/yr	Gal/yr		N	P_2O_5	K_2O	N	P_2O_5	K_2
				lb/ton			lb/1,000 gal		
Dairy	15	3614	12.7	10.0	4.1	7.9	41.5	17.0	32.8
Beef	11	2738	11.6	11.3	8.4	9.5	45.4	33.7	38.2
Veal	11.5	2738	8.4	8.7	2.1	9.0	36.5	8.8	37.8
Swine									
Growing pig	11.9	3008	9.2	13.8	10.8	10.8	54.6	42.7	42.7
Mature hog	5.9	1425	9.2	13.9	10.8	10.8	57.5	44.7	44.7
Sow & litter	15.9	3894	9.2	14.2	10.7	11.1	58.0	43.7	45.3
Sheep	7.3	1679	25.0	22.5	7.6	19.5	97.8	33.0	83.5
Goat	7.0	1789	31.7	22.0	5.4	15.1	86.1	21.1	59.1
Poultry									
Layers	9.7	2464	25.0	27.3	23.5	13.2	107.5	92.5	52.0

Figure A.2: *Cut-in cells have a hierarchical relationships to succeeding cells.*

Here s_0 substitutes for values similar to v_1 and v_2. This can be seen also in Figure A.1. In that case Through 1990 substitutes for information about particular events which are fixed in time and summarises a number of such 'hidden' events. The relationships between the identifier for the events (the names of the space missions) and the unique time at which each event occurs implies a possible relationships between the substituting contents and the category descriptions for the mission names and the number of crew

A.2.3 Orthogonal Domains, Under-Spans and L-Spans

Another peculiar arrangement of cells is what might be termed an **orthogonal domain.** This occurs when the author decides to place a distributed label cell group which causes a break in the tiling of the table. An alternative to this is to rotate the domain and create an over-spanned label.

The following is an example of an orthogonal domain:

(A.22)

Category	Composition Type	Distinguishing Features Chondrule Character	Letter Designation
Chrondites	Enstatite Chrondite	Distinct	E4
		Less Distinct	E5
		Characteristic Minerals	
Achrondites	Aubrites	Enstatite	AUB
	Angrite	Augite	ACANOM

The category `distinguishing features` has labels to cells which are marked by their spatial context.

Care should be taken when spans are considered. Occasionally, a group of cells which are aligned and which have the same value are merged to form a single cell which spans the cells which it is aligned with. Clearly, the fact that an equivalent value appears in a number of reading paths means that there is some vector of classification along which these are aligned, however this may be either:

- clarifying/aesthetic (particularly, reducing the amount of characters printed).
- an independent category.

In the following example, the under-span simply requires fewer characters in the table, and so clarifies to a certain extent.

(!A.23)

Vehicle		
Car	Train	Boat
Wheels		Props
4	64	4

This example (adapted from E18), however, has independent categories:

(A.24)

Nutrient Content			
Total N[b]	NH_4	P2OS	K_2O
1991			

This is similar to a cut-in but is a localised effect.

Finally, a rare though interesting case (P21) has the following physical form:

(A.25)

Historic Sites	Specially designated
732	49
796	51

which we term an l-span due to its shape.

A.2.4 Structure Templates

A rather frustrating interpretation of structure is the **structure template.** This arrangement indicates the relationship between cells not by using the above physical cues but by the high level cue of the template. In this case, the type of cells is indicated in an example (usually at the top of the table) and then subsequent repetitions of that physical layout are interpreted as containing cells inferior to those in the equivalent position in the template.

(A.26)

Rank	Movie Title		
	first weekend	first month	gross
1	Star Wars		
	100	200	500
2	ET: The Extraterrestrial		
	100	200	500

In the above, `Star Wars` and `ET: The Extraterrestrial` are inferior to `Movie Title`. In the following, a similar arrangement exists for the cells indicating examples of `constraints`.

(A.27)

name	type	predicate category
constraints		
M1	1, 2, 3	action
sub-clause:SEM: motivated=agent		
M2	1, 2, 3	23#23
sub-clause:SEM: motivated=experiencer		

A.3 Analysis

Now that the table has been given a more formal characterisation, it is perhaps useful to revisit the phenomena catalogued in Chapter 4 and to give them a full description in terms of this representation.

A.3.1 Recapitulation

(A.1)

$Animal_{cell_0}$			
$Horse_{cell_1}$		Pig_{cell_2}	
old_{cell_3}	$young_{cell_4}$	old_{cell_5}	$young_{cell_6}$
v_0	v_1	v_2	v_3

$$
\begin{aligned}
\mathcal{CON} = \{ & \\
& \{ \\
& \langle cat_0, \text{Animal}, \{\langle cat_1, \text{Horse}, \emptyset \rangle, \\
& \qquad \langle cat_2, \text{Pig}, \emptyset \rangle \\
& \qquad \} \\
& \rangle, \\
& \langle cat_3, \oslash, \quad \{\langle cat_4, \text{old}, \emptyset \rangle, \\
& \qquad \langle cat_5, \text{young}, \emptyset \rangle \\
& \qquad \} \\
& \rangle, \\
& \} \\
\} &
\end{aligned}
$$

which, as might be expected, is the same analysis as would be given for the table below.

(A.2)

	Animal	
	Horse	Pig
old	v_0	v_2
young	v_1	v_3

A.3.2 Over-Spanned Label

(A.28)

$States_{cell_0}$		ϵ_{cell_1}	b_{cell_2}
q_{cell_3}	$sequence_{cell_4}$	q_{cell_5}	qq_{cell_6}
	$probability_{cell_7}$	1.0_{cell_8}	$.2_{cell_9}$
$r_{cell_{10}}$	$sequence_{cell_{11}}$	$r_{cell_{12}}$	$qr_{cell_{13}}$
	$probability_{cell_{14}}$	$0.0_{cell_{15}}$	$.1_{cell_{16}}$

$$
\begin{aligned}
T^{struc}=\{ & \\
& \langle cell_0, cell_3, \emptyset\rangle,\ \langle cell_0, cell_{10}, \emptyset\rangle, \\
& \langle cell_3, cell_4, \emptyset\rangle,\ \langle cell_4, cell_5, \emptyset\rangle, \\
& \langle cell_4, cell_6, \emptyset\rangle,\ \langle cell_3, cell_7, \emptyset\rangle, \\
& \langle cell_7, cell_8, \emptyset\rangle,\ \langle cell_7, cell_9, \emptyset\rangle, \\
& \langle cell_{10}, cell_{11}, \emptyset\rangle, \langle cell_{11}, cell_{12}, \emptyset\rangle, \\
& \langle cell_{11}, cell_{13}, \emptyset\rangle, \langle cell_{10}, cell_{14}, \emptyset\rangle, \\
& \langle cell_{14}, cell_{15}, \emptyset\rangle, \langle cell_{14}, cell_{16}, \emptyset\rangle, \\
& \langle cell_1, cell_5, \emptyset\rangle,\ \langle cell_2, cell_6, \emptyset\rangle, \\
& \langle cell_1, cell_8, \emptyset\rangle,\ \langle cell_2, cell_9, \emptyset\rangle, \\
& \langle cell_1, cell_{12}, \emptyset\rangle, \langle cell_2, cell_{13}, \emptyset\rangle, \\
& \langle cell_1, cell_{15}, \emptyset\rangle, \langle cell_2, cell_{16}, \emptyset\rangle \\
\} &
\end{aligned}
$$

A.3.3 Cut-in

The cut-in was presented in two forms (Section A.2.1): the precedented and the unprecedented.

A.3.4 Precedented

(A.16)

Unchanged options	
`l`	Left adjusted column
`c`	Centered adjusted column
`r`	Right adjusted column
`p{width}`	Equivalent to \ `parbox[t]{width}`.
`{decl.}`	Suppresses inter-column space and inserts *decl.* instead
Changed options	
`|`	Defines a column of width *width*. Every entry will will be enlarged by the width of the line in contrast to the original definitions of LaTeX
New options	
⋮	

$$
\begin{aligned}
&\mathcal{T}^{relsem} = \{ \\
&\langle cat_0, \\
&\quad \varnothing, \\
&\quad \{ \\
&\quad\quad \langle cat_1, \text{Unchanged options}, \\
&\quad\quad\quad \{ \\
&\quad\quad\quad\quad \langle cat_2, \text{l}, \emptyset \rangle, \\
&\quad\quad\quad\quad \langle cat_3, \text{c}, \emptyset \rangle, \\
&\quad\quad\quad\quad \langle cat_4, \text{r}, \emptyset \rangle, \\
&\quad\quad\quad\quad \langle cat_5, \text{p}, \emptyset \rangle, \\
&\quad\quad\quad \} \\
&\quad\quad \rangle \\
&\quad\quad \langle cat_6, \text{Changed options}, \\
&\quad\quad\quad \{ \\
&\quad\quad\quad\quad \langle cat_7, \text{|}, \emptyset \rangle, \\
&\quad\quad\quad \} \\
&\quad\quad \rangle \\
&\quad \} \\
&\rangle
\end{aligned}
$$

Unprecedented, unrelated

(A.17)

value$_{cell_0}$
Usage of the English Articles(140 sentences, 380 nouns)$_{cell_1}$
correct$_{cell_2}$
reasonable$_{cell_3}$
partially correct $_{cell_4}$
incorrect$_{cell_5}$
% of correct
The Old Man with a Wen(104 sentences, 267 nouns)$_{cell}$
correct$_{cell_7}$
reasonable$_{cell_8}$
partially correct$_{cell_9}$
incorrect$_{cell_{10}}$
% of correct$_{cell_{11}}$
clan essay "TENSEI JINGO"(23 sentences, 98 nouns)$_{cell_{12}}$
correct$_{cell_{13}}$
reasonable$_{cell_{14}}$
partially correct$_{cell_{15}}$
incorrect$_{cell_{16}}$
% of correct$_{cell_{17}}$

average$_{cell_{18}}$
% of appearance$_{cell_{19}}$
% of correct$_{cell_{20}}$

$$
\begin{aligned}
T^{relsem} = \{ & \\
& \langle cat_0, \\
& \quad \text{value}, \\
& \quad \{ \\
& \quad \langle cat_1, \text{correct}, \emptyset \rangle, \\
& \quad \langle cat_2, \text{reasonable}, \emptyset \rangle, \\
& \quad \langle cat_3, \text{partially correct}, \emptyset \rangle, \\
& \quad \langle cat_4, \text{incorrect}, \emptyset \rangle, \\
& \quad \langle cat_5, \%\ \text{of correct}, \emptyset \rangle \\
& \quad \} \\
& \rangle \\
& \langle cat_6, \\
& \quad \text{average}, \\
& \quad \{ \\
& \quad \langle cat_7, \%\ \text{of appearance}, \emptyset \rangle,
\end{aligned}
$$

$\langle cat_8$, % of correct, $\emptyset$ $\rangle$
}
$\rangle$
$\langle cat_9$,
$\emptyset$,
{
$\langle cat_{10}$, Usage of the English Articles(140 sentences, 380 nouns), $\emptyset$ $\rangle$,
$\langle cat_{11}$, The Old Man with a Wen(104 sentences, 267 nouns),$\emptyset$ $\rangle$,
$\langle cat_{12}$, clan essay "TENSEI JINGO"(23 sentences, 98 nouns), $\emptyset$ $\rangle$
}
$\rangle$
}

Unprecedented, related

(A.18)

Animal Type$_{cell_0}$
Dairy$_{cell_1}$
Beef$_{cell_2}$
Veal$_{cell_3}$
Swine$_{cell_4}$
Growing Pig$_{cell_5}$
Mature Hog$_{cell_6}$
Sow & litter$_{cell_7}$
Sheep$_{cell_8}$
Goat$_{cell_9}$
Poultry$_{cell_{10}}$
Layers$_{cell_{11}}$

T^{struc}={
$\langle cell_0, cell_1, \emptyset\rangle$, $\langle cell_0, cell_2, \emptyset\rangle$,
$\langle cell_0, cell_3, \emptyset\rangle$, $\langle cell_0, cell_4, \emptyset\rangle$,
$\langle cell_4, cell_5, \emptyset\rangle$, $\langle cell_4, cell_6, \emptyset\rangle$,
$\langle cell_4, cell_7, \emptyset\rangle$, $\langle cell_4, cell_8, \emptyset\rangle$,
$\langle cell_4, cell_9, \emptyset\rangle$, $\langle cell_0, cell_{10}, \emptyset\rangle$,
$\langle cell_{10}, cell_{11}\rangle$
}

T^{relsem} ={
$\langle cat_0$, Animal Type,

{
⟨cat_1, Dairy, ∅ ⟩,
⟨cat_2, Beef, ∅ ⟩,
⟨cat_3, Veal, ∅ ⟩,
⟨cat_4,
Swine,
{
⟨cat_5, Growing Pig, ∅ ⟩,
⟨cat_6, Mature Hog, ∅ ⟩,
⟨cat_7, Sow & Litter, ∅ ⟩,
⟨cat_8, Sheep, ∅ ⟩,
⟨cat_9, Goat, ∅ ⟩,
}
⟨cat_{10}, Poultry, { ⟨cat_{11}, Layers,∅ ⟩ } ⟩
}
⟩
}

A.3.5 Orthogonal Domains

(A.29)

Category$_{cell_0}$	Composition Type$_{cell_1}$	Distinguishing Features$_{cell_2}$	Letter Designation$_{cell_3}$
		Chondrule Character$_{cell_4}$	
Chrondites$_{cell_5}$	Enstatite Chrondite$_{cell_6}$	Distinct$_{cell_7}$	E4$_{cell_8}$
		Less Distinct$_{cell_9}$	E5$_{cell_{10}}$
		Characteristic Minerals$_{cell_{11}}$	
Achrondites$_{cell_{12}}$	Aubrites$_{cell_{13}}$	Enstatite$_{cell_{14}}$	AUB$_{cell_{15}}$
	Angrite$_{cell_{16}}$	Augite$_{cell_{17}}$	ACANOM$_{cell_{18}}$

T^{struc} ={
⟨$cell_2, cell_4, \emptyset$⟩, ⟨$cell_4, cell_7, \emptyset$⟩,
⟨$cell_4, cell_9, \emptyset$⟩, ⟨$cell_2, cell_{11}, \emptyset$⟩,
⟨$cell_{11}, cell_{14}, \emptyset$⟩,⟨$cell_{11}, cell_{17}, \emptyset$⟩
}

Underspans

(A.30)

Vehicle$_{cell_0}$		
Car$_{cell_1}$	Train$_{cell_2}$	Boat$_{cell_3}$
Wheels$_{cell_4}$		Props$_{cell_5}$
4$_{cell_6}$	64$_{cell_7}$	4$_{cell_8}$

T^{struc} ={

$$
\begin{array}{ll}
\langle cell_0, cell_1, \emptyset\rangle, & \langle cell_0, cell_2, \emptyset\rangle, \\
\langle cell_0, cell_3, \emptyset\rangle, & \langle cell_1, cell_4, \emptyset\rangle, \\
\langle cell_2, cell_4, \emptyset\rangle, & \langle cell_3, cell_5, \emptyset\rangle \\
\langle cell_4, cell_6, \{cell_2\}\rangle, & \langle cell_4, cell_7, \{cell_3\}\rangle \\
\langle cell_5, cell_8, \emptyset\rangle &
\end{array}
$$

}

Appendix B

API

B.1 Overview

The core of the Tabpro system is implemented in C++. The API is presented below simply as the C++ class definitions taken straight from the source code.

B.2 Resource API

Creating a resource (Section 7.6) requires the implementation of three specific parts: the request object, the result object and the resource itself. Any resource acts by fielding a result object in response to a request object.

```
class ResourceRequest: public Object{

private:

  static int resourceRequestCounter;
  int id;

  ResourceTask task;

public:

  ResourceRequest(ResourceTask);
  virtual ~ResourceRequest() = 0;
  ResourceTask atask() const{return task;}
  int aid()const{return id;}

};
```

```
class ResourceRequestResult:public Object{

private:

  static int resourceRequestResultCounter;
  int requestId;
  int id;

public:

  ResourceRequestResult(int);
  virtual ~ResourceRequestResult() = 0;

};

class Resource: public Object{

  friend class ResourceManager;

private:

  String name;
  ResourceTask task;

public:

  Resource(const String &, ResourceTask);
  virtual ~Resource()=0;

public:

  virtual bool init()=0;
  virtual ResourceRequestResult *fieldRequest(ResourceRequest *) = 0;

  const String &aname() const{return name;}
  ResourceTask atask() const{return task;}

protected:

  fstream initFile;

  bool openInitFile();
```

```
  bool closeInitFile();

  bool getInitFileLine(String &);


};
```

B.3 Module API

The modules (Section 7.7) implemented by the system are identified by a name, a task and a floatingpoint number indicating the quality or belief in the results that the system has in the module (usually set to `1.0`). Modules must be capable of dealing with input and output from and to various files.

```
class Module: public Object{

protected:

  String name;
  String personalName;
  ModuleTask taskId;
  float confidence;

public:

  Module(const String &, ModuleTask, float);
  virtual ~Module()=0;

  virtual bool init(const Array &) = 0;
  virtual bool clear();
  ModuleTask itaskId();

  enum runMode{
    NORMAL,
    COMPILE
  };

  virtual bool run(runMode, const Array &)  = 0;

  const String &aname() const{return name;}
  void resetPersonalName();
```

```
    void assertModuleNameByTaskType();

    void getHtmlFileName(String &) const;

  protected:

    /*note that not all modules will require an output file*/

    fstream currentOutputFile;
    fstream currentCompileFile;
    fstream initFile;

    bool openOutputFile();
    bool closeOutputFile();

    bool openCompileFile();
    bool closeCompileFile();

    void output(float , runMode);
    void output(int , runMode);
    void output(const String &, runMode);
    void output(const char *, runMode);

    bool constructOutputFilePath(String &)const;
    bool constructCompileFilePath(String &)const;
    bool constructHTMLFilePath(String &)const;

    bool openInitFile();
    bool openInitFileW();
    bool closeInitFile();

    bool getInitFileLine(String &);

};
```

B.4 Hypothesis API

A hypothesis (Section 7.8) places an assertion in the pool of assertions managed by the hypothesis manager.

```
class Hypothesis:public Object{
```

```
private:

  //module information
  ModuleTask moduleTask;
  String moduleName;

  //session information
  long sessionStamp;

  //timeInformation
  long timeStamp;

  //hypothesis

  float confidence;
  Assertion *assertion;

public:
  Hypothesis(ModuleTask, const String &, long, long, float, Assertion *);
  ~Hypothesis();

  ModuleTask imoduleTask() const;
  void imoduleName(String &) const;

  friend int operator==(const Hypothesis &, const Hypothesis &);

  Assertion::assertionType itype() const;
  Assertion *iassertion() const;

};
```

B.5 Assertion API

An assertion (Section 7.8) is the basic element used to store results in the hypothesis manager.

```
class Assertion:public Object{

public:
```

```
enum assertionType{
  CELL,
  CELL_FUNCTION,
  CELL_STRUCTURE,
  TABLE_STRUCTURE,//the paths from a cell
  RELATIONAL_SEMANTICS_CATEGORY
};

  assertionType type;

public:

  Assertion(assertionType);
  virtual ~Assertion()=0;

  friend int operator==(const Assertion &, const Assertion &);
  virtual bool equal(const Assertion *) const =0;

};
```

Appendix C

Table Markup

C.1 Introduction

This appendix attempts to catalogue some of the markup formats currently in use for encoding tables. A number of the systems unearthed in the course of this research have only been found as references and all of the DTDs have not yet been found. However, in the interest of completeness, these cases are still included.

C.2 HTML

HTML([W3C98]) is the markup system currrently used describe documents published on the web. Originally a standalone system, it has been brought into the SGML fold and can now be described by a DTD.

C.2.1 Cells

Cells in the HTMLtable are marked up either as table head cells or table data cells. The distinction is to provide an indication to the viewer application that the head should or may be rendered in a bold or otherwise distinguished manner.

```
<!ELEMENT (TH|TD) - O %block>
```

A block expands as follows:

```
<!ENTITY % block "(%blocklevel|%inline)*">
```

blocklevel being:

```
<!ENTITY % blocklevel
        "P|%heading|%list|%preformatted|DL|DIV|CENTER|
```

```
        NOSCRIPT|NOFRAMES|BLOCKQUOTE|FORM|ISINDEX|HR|
        TABLE|FIELDSET|ADDRESS">
```

and inline being:

```
<!ENTITY % inline "#PCDATA|%font|%phrase|%special|%formctrl">
```

Headings are simply the different heading sizes (represented by the character 'h' and a digit between 1 and 6). A list is either an unordered list, an ordered list a DIR or a menu. preformatted is PRE.

`font` is simply the set of font wrappers (TT, I, B, U, A, ATRIKE, BIG, SMALL); `phrase` is another set of wrappers with a more content based feel (EM, STRONG, DFN, CODE, SAMP, KBD, VAR, CITE, ACRONYM); `special` expands to web type objects (A, IMG, APPLET, OBJECT, FONT, BASEFONT, BR, SCRIPT, MAP, Q, SUB, SUP, SPAN, BDO, IFRAME); and finally, `formctrl` are those tags associated with the appearance of forms (INPUT, SELECT, TEXTAREA, LABEL, BUTTON).

C.2.2 Grouping Cells

Grouping Rows The first basic way to group cells is through the `tablerow` ((TR)) tag. It collects cells horizontally as you might expect. It is context sensitive to the existance of other grouping strategies which may have appeared before. For example, spanning row cells.

Grouping Columns Columns can be grouped through the COLSPAN attribute of a TD or TH. There are other reasons to group columns, which don't result in alterations to the layout of the cells, but which provide a means of collectively describing certain features like alignment:

- **COL.**
- **COLGROUP.**

C.2.3 Gross Structure

At the top level, an HTML table has a caption (optional), column grouping instructions (zero or more), optional head and footer information and a body; the body (and the header and footer) contain table rows which in turn contain cells. The HTML description of tables is strongly row based, though it does allow mechanism for the merging of cells over rows and columns.

C.3 Text Encoding Initiative

The Text Encoding Initiative ([TEI95]) '*is an international project to develop guidelines for the preparation and interchange of electronic texts for scholarly research, and to satisfy a broad range of uses by the language industries more generally.*'

C.3.1 Cells

Cells are not all they seem to be in this markup. A row is used as in HTML to group 'cells' horizontally, however, these groups may contain cells or tables. In this respect the recursion happens in a slightly different manner to that of HTML. The cell tag can also contain a table, though it is not clear why this redundancy is included.

The cell tag may contain any of a number of general purpose elements found in the TEI definition body. As does the HTML cell, the TEI cell has row and column attributes presumably defining the possible spanning of rows and columns. There also exists a complete mechanism for referencing other cells (and rows) using IDs and IDREFs.

C.3.2 Grouping Cells

Rows Rows are formed by one ore more cells or tables.

C.3.3 Gross Structure

The gross structure is zero or more heads followed by one or more rows.

C.4 Exchange Table Model

The Exchange table model is suprisingly simple. However, it is hidden behind a lot of notational confusion. It is summarised in a slightly simplified form below (the simplification being the lack of exceptions).

```
<!ELEMENT table - O (title?, tgroup+)         >
<!ELEMENT tgroup- O (colspec*, thead?, tbody) >
<!ELEMENT colspec - O EMPTY >
<!ELEMENT thead - O (row+)    >
<!ELEMENT tbody - O (row+)    >
<!ELEMENT row   - O (entry+)  >
<!ELEMENT entry - O (#PCDATA*)>
```

Colspec is used to provide local information for the column it defines. This is generally formatting information. Spanning is provided by the entry attributes namest and nameend which indicate the columns at the left and right of the entry.

The names are references to the names of columns defined by the related attribute in colspec. Vertical spanning is provided by the morerows attribute. It states how many additional rows vertically the entry spans.

C.4.1 Gross Structure

Again, this is a row based model of tables. There is an additional twist in that the tgroup bundles parts of an overall table; it looks like you could realy create multiple/complex tables with this device.

C.5 Cameron's Model

Cameron offers a simple and clear content markup model ([Cam89]). The cells are marked declaritively by their logical (relative) position. This is very similar to Douglas and Hurst's (later) original model.

C.6 PHIGS Slide Set

The PHIGS ([Tho93b]) slide set adopts the table model based on MIL-M-28001A.

C.7 Air Transport Association

The ATA uses the MIL-M-28001A definition.

C.8 Association of American Publishers

Appears to use ISO 12083. The table model here seems very restricted: no support for spanning of rows and columns.

```
<!ELEMENT table          - -  (no?, title?, tbody)      -(%i.float;)  >
<!ELEMENT tbody          - O  (head*, tsubhead*, row*)                >
<!ELEMENT row            - O  (tstub?, cell*)                         >
<!ELEMENT tsubhead       - O  %m.ph;                                  >
<!ELEMENT (tstub|cell    - O  %m.pseq;                                >
```

C.9 Addison-Wesley

Nothing has yet been found about this specification.

C.10 DocBook

DocBook's ([Com00]) dtd version 3 says that it has 'changed over to the SGML Open full CALS table model'. The file included with the distribution (cals-tbl.dtd) appears to be very close to other table definitions seen managed by CALS. The definition includes much indirection (abstract). As before, the following summarizes the tag set.

```
<!ELEMENT table - - (title?, (tgroup+|graphic+) -(table|chart|figure))>
<!ELEMENT tgroup - O (colspec*, spanspec*, thead?, tfoot?, tbody)>
<!ELEMENT colspec - O EMPTY>
<!ELEMENT spanspec - O EMPTY>
<!ELEMENT (thead|tfoot) - O (colspec*, row+) -(entrytbl)>
<!ELEMENT tbody - O (row+)>
<!ELEMENT row - O ((entry|entrytbl)+) -(pgbrk)>
<!ELEMENT entrytbl - - (colspec*, spanspec*, thead?, tbody?)
                                                -(entrytbl|pgbrk)>
<!ELEMENT entry - O ((para|warning|caution|
                                    note|legend|#PCDATA)*) -(pgbrk)>
```

C.11 Exoterica Complex Tables

C.12 ISO/IEC TR 9573-11

C.13 MIL-M-28001A

C.14 SoftQuad

According to the PHIGS document, this is another row based markup model.

C.15 Douglas-Hurst Model

Similar in style to Cameron.

C.16 LaTeX

The basic tabular environment in LaTeXprovides a row based model. Spanning of columns is provided by the multicolumn macro which uses relative parameters to specify the span.

System	Reference	Mentioned in
LaTeX	[Lam85]	
SoftQuad		[Tho93a]
Exoterica Complex Tables		
Phigs	[Tho93a]	
TEI	[TEI95]	
HTML	[W3C98]	
MIL-M-28001A		[mar99b]
Exchange Table Model		
Cameron	[Cam89]	
Air Transport Association		[mar99b]
Association of American Publishers		[mar99a]
Adison Wesley		
DocBook	[Com00]	
CALS	[Div00]	
ISO 12083		[mar99a]

Figure C.1: A summary of markup resources for tables.

C.17 Summary

All of the above use a row based markup method, except Cameron and D-H.

Appendix D

Table Processing Workbench Manual

`cleardocs` : clear all the documents from the document manager.

`clearhyps` MODULE : clear any hypotheses registered by MODULE.

`clearmod` MODULE : clear any settings that MODULE may have, effectively resetting.

`compile` MODULE *module_arguments* : run the name module with the module dependent arguments and produce the appropriate compile information.

`corpus` *file_name* :

`define` *@variable_name string* :

`document` *file_name* :

`exit` : exit from the table processing workbench.

`html` *file_name* : produce an HTML version of the current table and write it to *file_name*.

`initmod` MODULE : (re-)initialise MODULE.

`latex` *file_name* : produce a LaTeX version of the current table and write it to *file_name*.

`ldocs` : list the documents registered with the document manager.

`loadfunc` : load in the functional description of the table from the files with the suffix `.cell`.

`ltabs` : list the tables found in the current document.

`mode` *mode_name* : set the mode of the system output to *mode_name*. Currently this can only be *html_mode* which indicates that the system should output a trace of the processes in an HTML format.

`resource` RESOURCE : register RESOURCE with the resource manager.

`module` MODULE : register MODULE with the module manager.

`run` MODULE *module_arguments* : run the named module with the module dependent arguments. Each module has a different set of arguments, however there are a number which are common to most modules.

-p private_name :

-h hypotheses_constraints :

`setdoc` *doc_name* : set the current document to be the document registered with the name *doc_name*.

`settable` *table_name* : set the current table to be the table registered with the name *table_name*.

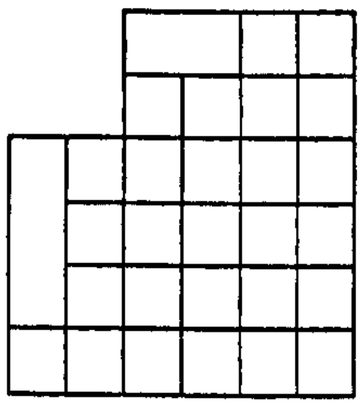
`shell` *shell_line* : interpret the arguments using the operating systems shell running the table processing workbench.

Appendix E

Algorithms

The algorithm is presented with an inline example demonstrating which cells from the example table given below, are being considered by the algorithm at most of the major steps.

(E.1)

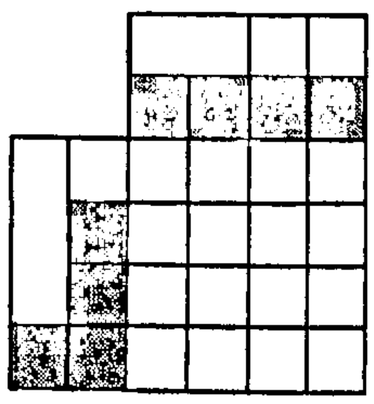


$\forall c \in Tab$

if($c \in \mathcal{A}$){

/*if there is a unique adjacent cell above*/

if($\uparrow c \uparrow = 1$ and not a cut in cell and not a repeated cut in){



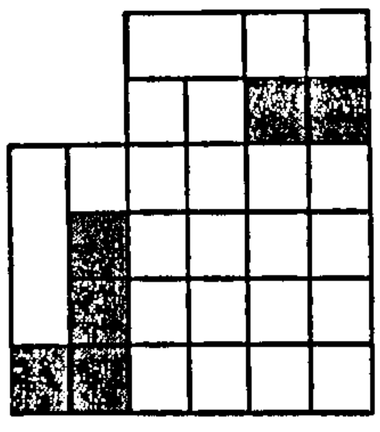
$tempCell \leftarrow c[_{top}0]$

$store \leftarrow 0$

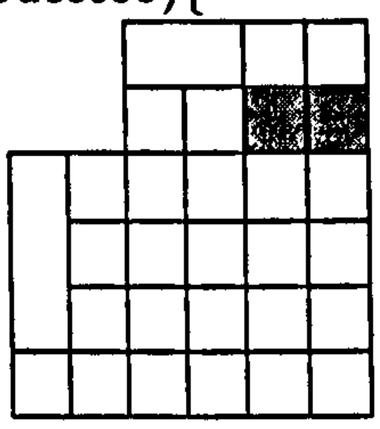
$success \leftarrow true$

/*if that cell is perfectly aligned*/

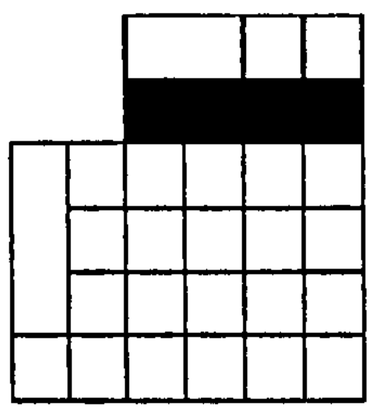
if($tempCell \Updownarrow c$ and $tempCell$ is not a cut in){



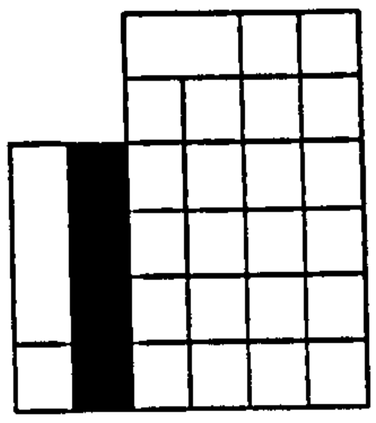
```
/*look up incase there is a cut in cell*/
while(tempCell ⇳ c and tempCell is not a cut in){
    if(↑ tempCell ↑ = 1){
        store ← tempCell
        tempCell ← tempCell[top 0]
    }/*end if*/
    else success←false
}/*end while*/
if(tempCell ⇳ c and tempCell is not a cut in){
    success←false
}/*end if*/
}/*end if*/
if(success){
```



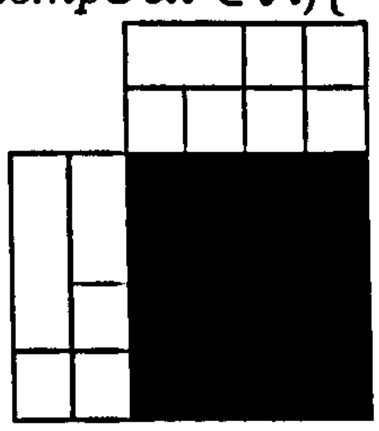
```
if(tempCell ∈ A){
    if(tempCell ∩ c or tempCell is a cut in){
        /*check left for spanning*/
        test ← false
        if(←c = 1){
            left ← c[left 0]
            if(↑ left ↑= 1 and →left > 1){
                if(left[top 0] = →tempCell){
                    test ← true
                }/*end if*/
            }/*end if*/
        }/*end if*/
        if(test = false){
```



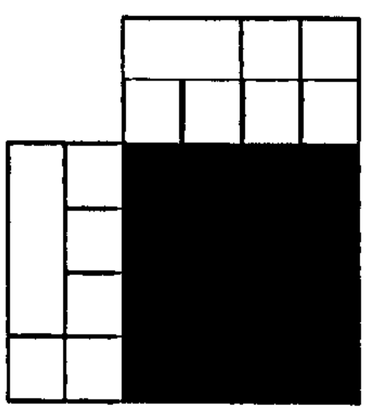
creat str link from tempCell to c
}/*end if*/
if(*store* and $\neg$*store* = *c*){
if(all store daughters are $\mathcal{A}$){
create str link from store to c
}/*end if*/
if(*tempCell* is a cut in){
store $\leftarrow$ uppermost cell which is perfectly aligned with c
create str link store c
}/*end if*/
}/*end if*/
}/*end if*/
}/*end if*/
else{
if(c is not a cut in){
$\forall$ the daughters (*d*) above *c*
tempCell $\leftarrow$ $c[_{top}d]$
if(*tempCell* $\in \mathcal{A}$){
create str link tempCell c
}/*end if*/
else{
store $\leftarrow$the upper left most cell left aligned with *c*
create an str link from store to c
}/*end if*/
if($\overleftarrow{c} = 1$){
tempCell $\leftarrow$ $c[_{left}0]$
}
if(tempCell $\in \mathcal{A}$){
if(tempCell $\leftrightarrow$ c and tempCell is not a cut in){



create str link tempCell c
}/*end if*/
}
}
if($c \in \mathcal{D}$){
$tempCell \leftarrow c$
$\overleftarrow{tempCell}$
while($\overleftarrow{tempCell} = 1$ and $tempCell \in \mathcal{D}$){
$\overleftarrow{tempCell} \leftarrow tempCell[_{left}0]$
}
if($tempCell \in \mathcal{A}$){



create str link tempCell c
}
$tempCell \leftarrow c$
while($\uparrow tempCell \uparrow = 1$ and tempCell $\in \mathcal{D}$ or tempCell is a cut in)
if($\uparrow tempCell \uparrow > 1$ or $tempCell$ is a cut in){
$tempCell \leftarrow$ perfectly vertically aligned cell above
}
else{
$tempCell \leftarrow$ tempCell top
}
}
if($tempCell \in \mathcal{A}$){



```
    create str link tempCell c
}
```

Appendix F

Semantic Grammar

The following description is of a regular grammar used to recognise certain spans of text which have a simple semantic tag.

```
$some basic number expressions to start off with

_DIGIT: 0|1|2|3|4|5|6|7|8|9

_NATURAL-NUMBER: _DIGIT.(_DIGIT?._DIGIT)?.((,?._DIGIT._DIGIT._DIGIT)+)?

_INTEGER-NUMBER: (-|#+)?._NATURAL-NUMBER

_RATIONAL-NUMBER: _NATURAL-NUMBER?.#.._DIGIT+

@_NUMBER: _INTEGER-NUMBER|_RATIONAL-NUMBER

@_NUMBER-CLASS: number|(number.of)|Number|(Number.of)|no|(no#.)|No|(No#.)|
                ##|(##.of)

@_CABLE-TYPE: Cable.type

$$$  ARITHMETIC PATTERNS

@_FRACTION-NUMBER: _NUMBER./._NUMBER

@_RANGE-NUMBER: _NUMBER.(-|to)._NUMBER

@_RATIO-NUMBER: _NUMBER.:._NUMBER

@_PERCENT: percent|percentage|Percent|Percentage|%
```

```
@_PERCENTAGE-NUMBER: (_NUMBER|_RANGE-NUMBER).#(?._PERCENT.#)?

$$$expressions of quantity

_STANDARD-UNIT:  mile|miles|m|\
                 kilometer|kilometre|kilometers|kilometres|km|\
                 meter|metre|meters|metres|m|\
                 centimeter|centimetre|centimeters|centimetres|cm|\
                 millimeter|millimetre|millimeters|millimetres|mm|\
                 micrometer|micrometre|micrometers|micrometres|\
                 nanometer|nanometre|nanometers|nanometres|nm|\
                 angstrom|angstroms|A|\
                 inch|inches|in|\
                 foot|feet|ft|\
                 centigrade|o?C|\
                 o?F|\
                 kelvin|K|\
                 gram|gramme|grams|grammes|g|\
                 ounce|ounces|oz|\
                 kilogram|kilograms|kg|kgs|\
                 ton|tonne|tons|tonnes|Ton|Tonne|Tons|Tonnes|\
                 pound|pounds|lb|lbs|\
                 gallon|gallons|gal|gals|Gallon|Gallons|Gal|Gals|\
                 microsecond|microseconds|us|\
                 milliseconds|millisecond|ms|\
                 seconds|s|\
                 minutes|minute|mins|min|\
                 hours|hour|hrs|hr|Hrs|Hr|\
                 day|days|\
                 week|weeks|\
                 month|months|\
                 year|years|yr|yrs\
                 decade|decades|\
                 century|centuries|\
                 millenium|millenia|\
                 eon|eons|\
                 ohm|ohms|O|\
                 hertz|Hz|\
                 kilohertz|kHz|\
                 volt|volts|V|\
```

```
electron.volt|electron.volts|eV|\
uH|\
watts|W|\
kilowatts|kW\
degree|degrees|deg

_POWER-UNIT: _STANDARD-UNIT.^._DIGIT+

_UNIT-RATIO: ((_STANDARD-UNIT|_POWER-UNIT)./.(_STANDARD-UNIT|_POWER-UNIT))|\
                ((_STANDARD-UNIT|_POWER-UNIT)./._INTEGER-NUMBER._STANDARD-UNIT)

@_UNIT:(_STANDARD-UNIT|_POWER-UNIT|_UNIT-RATIO)|(_NUMBER-CLASS)

@_EXPLICIT-QUANTITY: (_NUMBER|_RANGE-NUMBER).#(?.(_UNIT).#)?

@_PAREN-UNIT: #(._UNIT.#)

$expressions of currency

_CURRENCY: $|Y|lb|lbs

@_MONETARY-AMOUNT: _CURRENCY._NUMBER

$expressions of time

_YEAR: (((1|2)._DIGIT)|')?._DIGIT._DIGIT

_DAY-OF-WEEK: monday|tuesday|thursday|friday|saturday|sunday|\
                mon|tues|wed|thur|thurs|fri|sat|sun

_CLOCK: _DIGIT._DIGIT?.:?._DIGIT._DIGIT

_DAY-OF-MONTH: _DIGIT|((0|1|2|3)._DIGIT)

_DAY-OF-MONTH-ORDINAL:
((1|21|31).st)|((2|22).nd)|((3|23).rd)|((4|5|6|\
                7|8|9|10|11|12|13|14|15|16|17|18|19|20|24|25|26|27|28|29|30).th)

_MONTH: january|february|march|april|may|june|july|august|september|october\
                |november|december
```

```
_MONTH-CARDINAL: 1|2|3|4|5|6|7|8|9|10|11|12|\
                 01|02|03|04|05|06|07|08|09


@_DATE: (_DAY-OF-MONTH./._MONTH-CARDINAL./._YEAR)|\
        (_MONTH-CARDINAL./._DAY-OF-MONTH./._YEAR)|\
        (_MONTH-CARDINAL./._YEAR)|\
        (_DAY-OF-WEEK.the._DAY-OF-MONTH-ORDINAL.of._MONTH._YEAR?)|\
        (_MONTH.the?._DAY-OF-MONTH-ORDINAL.,?._YEAR)|\
        (_DAY-OF-MONTH.-._MONTH.-._YEAR)|\
        _YEAR
```

Appendix G

Notes on the development corpus

Nearly all the tables used to develop the model and to train and test the IE application are real examples found in published documents. In certain cases in the thesis, in order to make a point in a concise and relevant manner, tables and table fragments have been constructed.

Examples were drawn from the following publications: [GBB91a], [Mag97], [New97], [Cha96], [Lag73], [oE86], [Dew25], [Miu86], [LeB97], [TS97], [Med99].

The remainder of the examples, and the corpus for the development and testing of the implemented system came from the following set of web pages.

- `http://www.hastings.edu/resource/career/cs012.htm`
- `http://www.ipc.on.ca/web_site.eng/locating/orders-p/P-961.htm`
- `http://www.hq.usace.army.mil/cemp/e/es/aesurvey.htm`
- `http://www.spirit.com.au/Dreaming/bikes1.htm`
- `http://www.ozemail.com.au/~ieinfo/ooie.htm`
- `http://www.cis.ufl.edu/~georges/cis4301/schedule.htm`
- `http://www.genweb.com/Dnavax/Patents/5620896.html`
- `http://www1.cc.emory.edu/MOLECULAR_VISION/instructions.html`
- `http://www.emory.edu/molvis/v1/wistow/index.html`
- `http://www.barra.com/ResearchPub/BarraPub/pmac-n.html`
- `http://www.csc.calpoly.edu/~dstearns/315/Manual/microMachine.html`

- http://www.virusbtn.com/VBPapers/Ivpc96/index.html
- http://www.cv.nrao.edu/aips/ddt.html
- http://www.kai.com/benchmarks/stepanov/index.html
- http://www.nist.gov/itl/div894/894.01/proc/darpa97/html/seymore1/seymore1.htm
- http://image-gw.esys.tsukuba.ac.jp/html/yuichi/acm97/main.html
- http://www.alaska.net/~meteor/type.htm
- http://www.ag.ohio-state.edu/~ohioline/b604/b604_15.html
- http://www-aghort.massey.ac.nz/departs/soilsc/cybsoil/ruapehu/ruapehu.htm
- http://www.osf.hq.nasa.gov/shuttle/futsts.html
- http://www.osf.hq.nasa.gov/spacemen.html
- http://www-nsidc.colorado.edu/NASA/GUIDE/docs/dataset_documents/dmsp_ssmi_brightness_temperatures_and_sea_ice_concentration.html
- http://www.gsrg.nmh.ac.uk/~phoh/iqual1.htm
- http://www.mad-cow.org/~tom/Aug27_News.html
- http://www.kcmetro.cc.mo.us/longview/socsci/philosophy/logic/ttbl2.htm
- http://www.olympus.co.jp/LineUp/Digicamera/c1400lE.html
- http://www-osma.lerc.nasa.gov/lsm/lsm1.htm
- http://www.engr.orst.edu/~ullman/what1.htm
- http://www.dai.ed.ac.uk/students/timt/papers/twin_studies/studies.html
- http://www.hinet.cs.ritsumei.ac.jp/~ken/bachelor/Constructioneng.html
- http://www2.inter.co.jp/Baseball/f_97.html

Bibliography

[Bea85] R. J. Beach. *Setting Tables and Illustrations with Style.* PhD thesis, University of Waterloo, 1985. Also issued as Technical report CSL-85-3, Xerox Palo Alto Research Center, Palo Alto, CA.

[Bea86] Richard J. Beach. Tabular typography. In *Text Processing and Document Manipulation, Proceedings of the International Conference*, pages 18–33. The British Computer Society, Cambridge Univerity Press, 1986.

[BEF84] T. J. Biggerstaff, D. M. Endres, and I. R. Forman. Table: Object oriented editing of complex structures. In *Proceedings - International Conference on Software Engineering.* IEEE Computer Society, 1984.

[Bra85] Bradshaw. *Bradshaw's July 1922 Railway Guide.* Guild Publishing, 1985.

[Cam89] James P. Cameron. A cognitive model for tabular editing. Technical Report OSU-CISRC-6/89-TR 26, Computer and Information Science Research Center, Ohio State University, 1989.

[CB62] C. H. Corllis and W. R. Bozman. Experimental probabilities for spectral lines of seventy elements. Technical Report 53, NBS, 1962.

[CGJ+93] Jim Cowie, Louise Guthrie, Wang Jin, Rong Wang, and Takahiro Wakao. Crl/brandeis description of the *DIDEROT* system as used for muc-5. In *Proceedings of the fifth message understanding conference.*, 1993.

[CGW95] Hamish Cunningham, Robert J. Gaizauskas, and Yorick Wilks. A general architecture for text engineering (gate) - a new approach to language r & d. Technical Report CS-95-21, University of Sheffield, Institute for Language, Speech and Hearing (ILASH), and Department of Computer Science, University of Sheffield, UK, 1995.

[Cha96] Eugene Charniak. *Statistical Language Learning.* Language, Speech and Communication Series. MIT Press, 1996.

[CHL93] N. Chinchor, L. Hirschman, and D. D. Lewis. Evaluating message understanding systems: An analysis of the third message understanding conference (muc-3). *Computational Linguistics*, 19(3):409–449, 1993.

[CK93] Surekha Chandran and Rangachar Kasturi. Structural recognition of tabulated data. Technical Report TR-93-124, Computer Engineering Program, Department of Electrical and Computer Engineering, The Pennsylvania State University, University Park, Pennsylvania 16802, April 1993.

[CL96] Jim Cowie and Wendy Lehnert. Information extraction. *Communications of the ACM*, 39(1), January 1996.

[Col96] Robin Collier. Automatic template creation for information extraction, an overview. Technical Report CS-96-07, University of Sheffield, Department of Computer Science, 1996.

[Com00] DocBook Technical Committee. Docbook. http://www.oasis-open.org/docbook/, 2000.

[COO93] Lynn Carlson, Boyan Onyshkevych, and Mary Ellen Okurowski. Corpora and data preparation. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, 1993.

[Cor91] Lotus Development Corporation. *Improv Handbook.* Cambridge, MA, 1991.

[Cro95] Richard Crouch. Ellipsis and quantification: A substitutional approach. In *Proceedings of the 7^{th} European ACL*, Ireland, February 1995.

[DDMR90] Steven J. DeRose, David G. Durand, Elli Mylonas, and Allen H. Renear. What is text, really? *Journal of Computing in Higher Education*, 1(2):3–26, 1990.

[Dew25] Davis R. Dewey. American economic review, 15, 1925.

[DHQ95] Shona Douglas, Matthew Hurst, and David Quinn. Using natural language processing to interpret tables in plain text. In *Fourth Symposium on Document Analysis and Information Retrieval*, 1995.

[Div00] Navy Surface Warfare Center Division. Continuous acquisition and life-cycle support (cals). http://navycalls.dt.navy.mil, 2000.

[Fel99] Christiane Fellbaum. *WordNet: an electronic lexical database.* MIT Press, 1999.

[GBB91a] Guthrie, Britten, and Barker. Cognitive process of search. *Reading Research Quarterly*, 26(3), 1991.

[GBB91b] John T. Guthrie, Tracy Britten, and K. Georgene Barker. Roles of document structure, cognitive strategy, and awareness in searching for information. *International Reading Association*, 1991.

[GHH+82] B. Grosz, N. Haas, G. Hendrix, J. Hobbs, P. Martin, R. Moore, J. Robinson, and S. Rosenschein. Dialogic: A core natural-language processing system. In *COLING 82*, 1982.

[GK95a] E. Green and M. Krishnamoorthy. Model-based analysis of printed tables. In *Proceedings of International Conference on Document Analysis and Recognition 95*, pages 214–217, 1995.

[GK95b] E. Green and M. Krishnamoorthy. Recognition of tables using table grammars. In *Fourth Annual Symposium on Document Analysis and Information Retrieval*, pages 261–277, 1995.

[Goe87] Philip W. Goetz, editor. *The History of Mathematics*, volume 23. Encyclopedia Brittanica Inc., 1987. pages 612-614.

[Gol90] Charles F. Goldfarb. *The SGML Handbook*. Clarendon Press, Oxford, 1990.

[Gre97] Edward A. Green. Ph.d. research. http://tardis.union.edu/ greene/research-dir/research.html, 1997.

[Gro99] Language Technology Group. LTCHUNK. http://www.ltg.ed.ac.uk, 1999.

[GS96] R. Grishman and B. Sundheim. Message understanding conference-6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 466–471, Copenhagen, June 1996.

[Gut97] John T. Guthrie. Definition of category. Personal Communication, November 1997.

[GW98] Robert Gaizauskus and Y. Wilks. Information extraction: Beyond document retrieval. *Journal of Documentation*, 54(1):70–105, 1998.

[GWH+95] R. Gaizauskas, T. Wakao, K. Humphreys, H. Cunningham, and Y. Wilks. University of sheffield: Description of the lasie system as used for muc-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 207–220. Morgan Kaufman, 1995.

[GWK93] John T. Guthrie, Shelley Weber, and Nancy Kimmerly. Searching documents: Cognitive processes and deficits in understanding graphs, tables, and illustrations. *Contemporary Educational Psychology*, 18:186–221, 1993.

[HAB+] Jerry R. Hobbs, Douglas Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel, and Mabry Tyson. Fastus: A cascade finite-state transducer for extracting information from natural-language text. http://www.ai.sri.com/natural-language/projects/fastus-schabes.html.

[Ham95] Eric M. Hammer. *Logic and Visual Information.* CSLI Publications & folli, 1995.

[HD95] Osamu Hori and David S. Doermann. Robust table-form structure analysis based on box-driven reasoning. In *Proceedings of International Conference on Document Analysis and Recognition 95*, pages 218–221, 1995.

[HD97] Matthew Hurst and Shona Douglas. Layout and language: Preliminary experiments in assigning logical structure to table cells. In *Proceedings of ANLP-97*, 1997.

[HDG00] K. Humphreys, G. Demetriou, and R. Gaizauskas. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proceedings of the Pacific Symposium on Biocomputing (PSB-2000)*, pages 505–516, Honolulu, Hawaii, USA, January 2000.

[Hea98] Marti Hearst. *WordNet: An Electronic Lexical Database*, chapter Automated discovery of wordnet relations. MIT Press, Cambridge, MA, USA, 1998.

[Hea99] Marti A. Hearst. Untangling text data mining. In *37th Annual Meeting of the Association for Computational Linguistics*, Maryland, USA, June 1999.

[HKS96] Udo Hahn, Manfred Klenner, and Klemens Schnattinger. Automated knowledge aquisition meets metareasoning: Incremental quality assesment of concept hypotheses during text understanding. In B Gains and M Musen, editors, *KAW'96 - Proceedings of the 10th Banff Knowledge Aquisition for Knowledge-Based Systems WOrkshop*, 1996.

[Hob91] Jerr R. Hobbs. Sri international: Description of the tacitus system as used for muc-3. In *The Third Message Understanding Conference (MUC-3)*, 1991.

[Hob93] Jerry R. Hobbs. The generic information extraction system. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, 1993.

[Hoo90] J. T. Hooker. *Reading the Past.* British Museum Publications, 1990.

[Hur99a] Matthew Hurst. Layout and language: A corpus of documents containing tables. In *AAAI Fall Symposium on Using Layout for the Generation, Understanding and Retrieval of Documents.* AAAI, 1999.

[Hur99b] Matthew Hurst. Layout and language: Beyond simple text for information interaction. In *The 2nd International Conference on Multi-model Interfaces*, 1999.

[KD94] Alistair Knott and Robert Dale. Using linguistic phenomena to motivate a set of rhetorical relations. *Discourse Processes*, 18(1):35–62, 1994.

[KD98] T. Kieninger and Andreas Dengel. A paper-to-html table converting system. In *Proceedings of Document Analysis Systems (DAS) 98*, Nagano, Japan, November 1998.

[KW98] William Kornfeld and John Wattecamps. Automatically locating, extracting and analyzing tabular data. In *SIGIR '98. Proceedings of the 21st annual international ACM SIGIR conference*, pages 347–348, August 1998.

[Lag73] Lagrange. *Oeuvres De Lagrange.* Gauthier-Villars, 1773.

[Lam85] Leslie Lamport. *LaTeX User's Guide and Reference Manual.* Addison-Wesley, 1985.

[LeB97] LeBeau. Corticotrph action potential model. *Biophysical Journal*, 73(3), 1997.

[Lef89] Paul Lefrere. Design aids for constructing and editing tables. Technical Report 61, British Library Research and Development Department, 1989.

[LMS+93] W. Lehnert, J. McCarthy, S. Soderland, E. Riloff, C. Cardie, J. Peterson, F. Feng, and C. Dolan. Umass/hughes: Description of the circus system used for muc-5. In *Proceedings of the Fifth Message Understanding Conference*, pages 277–201, 1993.

[LN99] Daniel Lopresti and George Nagy. Automated table processing: An (opinionated) survey. In *The Third IAPR International Workshop on Graphics Recognition(GREC '99)*, 1999.

[LP95] I. Lewin and S. G. Pulman. Inference in the resolution of ellipsis. In *Proceedings of ESCA Research Workshop on Spoken Dialogue Systems*, March 1995.

[LS91] Wendy Lehnert and Beth Sundheim. A performance evaluation of text analysis technologies. *AI Magazine*, Fall 1991.

[LV92] A. Laurentini and P. Viada. Identifying and understanding tabular material in compound documents. In *International Conference on Pattern Recognition*, 1992.

[Mag97] PC Magazine. First looks, August 1997.

[mar99a] Iso 12083. http://www.mcs.net/ dken/i12083.htm, 1999.

[mar99b] Mil-m-28001a. http://navysgml.dt.navy.mil/cals.html, 1999.

[Med99] Math Medics. http://www.sosmath.com, 1999.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[Miu86] Yuki Miura. *Japan Statistical Year Book 1986.* Statistics Bureau Management and Coordination Agency, 1986.

[MTT+97] David McKelvie, Henry Thompson, Richard Tobin, Chris Brew, and Andrei Mikheev. *The Normalised SGML Library LT NSL Version 1.5.* Language Technology Group, Human Communication Research Centre, The University of Edinburgh, 1997.

[MUC95] MUC. Information extraction task definition. ftp://ftp.muc.saic.com/pub/MUC/, August 1995.

[New97] Newsweek. Cracking down on crime. Newsweek, June, 9 1997.

[Niy94] Debashish Niyogi. *A Knowledge-Based Approach To Deriving Logical Structure From Document Images.* PhD thesis, SUNY, Buffalo, New York, August 1994.

[NLK99] Hwee Tou Ng, Chung Yong Lim, and Jessica Li Teng Koo. Learning to recognize tables in free text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 443–450, Maryland, USA, June 1999.

[oE86] Actuarial Society of Edinburgh. *Actuarial Society of Edinburgh: Transactions.* Charles and Edwin Layton, London, 1886.

[PC97] Pallavi Pyreddy and W. Bruce Croft. Tintin: A system for retrieval in text tables. Technical Report 105, Center for Intelligent Information Retrieval, 1997.

[Pub86] Association Of American Publishers. Markup of tabular material. Technical report, Association of American Publishers, 1986. Manuscript Series.

[Pul94] Stephen G. Pulman. A computational theory of context dependence. In *Tilburg Workshop on Computational Semantics*, November 1994.

[RKG95] Klaus Reichenberger, Thomas Kamps, and Gene Golovchinsky. Towards a generative theory of diagram design. In *Proceedings of 1995 IEEE Symposium on Information Visualization*, pages 217–223, Los Alamitos, 1995. IEEE Computer Society Press.

[RR99] Tony Robinson and Steve Renals, editors. *Proceedings of the ESCA ETRW Workshop*, 1999.

[RS94] Daniela Rus and Kristen Summers. Using white space for automated document structuring. Technical Report TR94-1452, Cornell University, Department of Computer Science, July 1994.

[SBW97] John H. Shamillian, Henry S. Baird, and Thomas L. Wood. A retargetable table reader. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 158–163, August 1997.

[SP92] T. Saitoh and T. Pavlidis. Document image segmentation and text area ordering. In *11th International Conference on Pattern Recognition*, pages 277–280, The Hague, 1992.

[sty93] *The Chicago Manual of Style.* The University of Chicago Press, 14 edition, 1993.

[TAH+] Mabry W. Tyson, Douglas Appelt, Jerry R. Hobbs, John Bear, David Israel, and Megumi Kameyama. Recognizing and interpreting tables (*unpublished*).

[TEI95] TEI. The text encoding inititative. http://www.uic.edu/orgs/tei, June 1995.

[Tho93a] R. E. Thomas. Sgml tables for the phigs slide set. Technical Report RAL-93-029, Rutherford Appleton Laboratory, Chilton, Didcot, 1993.

[Tho93b] R. E. Thomas. Sgml tables for the phigs slide set. Technical report, Rutherford Appleton Laboratory, 1993.

[Tho96] Marcy Thompson. A tables manifesto. In *Proceedings of SGMK Europe*, pages 151–153, Munich, Germany, 1996.

[TS97] J. Takahashi and S. Sagayama. Vector-field-smoothed bayesian learning for fast and incremental speaker/telephone-channel adaptation. *Computer Speech and Language*, 11(2):127–146, April 1997.

[Ull88] Jeffrey D. Ullman. *Principles of Database and Knowledge-Base Systems.* Computer Science Press, 1988.

[Van92] C. Vanoirbeek. Formatting structured tables. In C. Vanoirbeek and G. Coray, editors, *EP92: Proceedings of Electronic Publishing*, UK, 1992. Cambridge University Press.

[W3C98] W3C. Html 4.0 specification. http://www.w3.org/TR/REC-html40/, April 1998.

[Wan96] Xinxin Wang. *Tabular Abstraction, Editing, and Formatting.* PhD thesis, University of Waterloo, Waterloo, Ontario, Canada, 1996.

[WBMT19] Ian H. Witten, Zane Bray, Malika Mahoui, and Bill Teahan. Text mining: A new frontier for lossless compression. In *UNKNOWN*, 19??

[WF70] Patricia Wright and Katheryn Fox. Presenting information in tables. *Applied Ergonomics*, 1:234–242, 1970.

[WF72] P. Wright and K. Fox. Explicit and implicit tabulation formats. *Ergonomics*, 1972.

[WHL84] P. Wright, A. J. Hull, and A. Lickorish. Psychological factors in reading tables. In *22nd International Conference on Psychology*, 1984.

[Wil97] Yorick Wilks. Information extraction as a core language technology: What is ie? Technical Report CS-97-15, University of Sheffield, Department of Computer Science, 1997.

[WLS93] T. Watanabe, Q. Luo, and N. Sugie. Toward a practical document understanding of table-form documents: Its framework and knowledge representation. In *Proceedings of the 2nd International Conference on Document Analysis and Recognition*, pages 510–515, 1993.

[WLS94] Toyohide Watanabe, Qin Luo, and Noboru Sugie. Knowledge for understanding table-form documents. *IEICE Transactions on Information and Systems*, E77-D(7), 1994.

[Woo] Derick Wood. Personal communication. email.

[Woo68] F. P. Woodford, editor. *Scientific Writing for Graduate Students.* Rockefeller University, New York, 1968.

[Wri68] P. Wright. Using tabulated information. *Ergonomics*, (11):331–343, 1968.

[Wri82] P. Wright. A user-oriented approach to the design of tables and flowcharts. In *The technology of Text, Principles for Structuring, Designing, and Displaying Text.* Educational technology Publications, June 1982.

[WW93] Xinxin Wang and Derick Woods. Tabular abstraction for tabular editing and formatting. In *3rd International Conference for Young Computer Scientists*, 1993.

[ZM95] P Zweigenbaum and Consortium MENELAS. Menelas: coding and information retrieval from natural language patien discharge summaries. In M. F. Laires, M. J. Laferira, and J. P. Christensen, editors, *Advances in Health Telematics*, pages 82–89. IOS Press, Amsterdam, 1995.

Term Index

The following index entries employ bold face to indicate the page where a term is defined.

Author Index

System Index

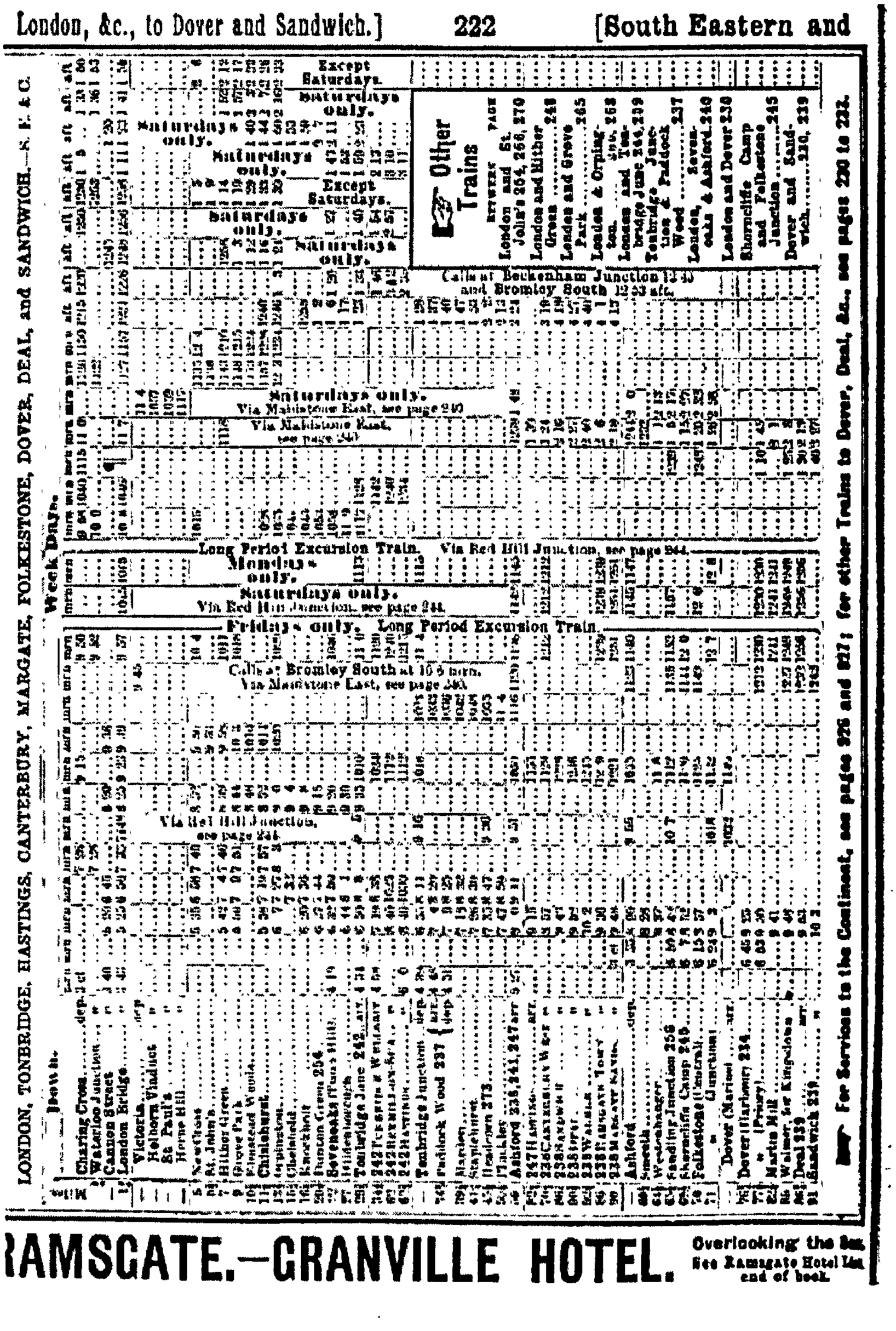


Figure G.1: A 1922 UK Train Timetable from [Bra85].